Closed-loop control of complex systems using deep Reinforcement Learning

An off-policy approach

T. Guégan[3], M.A. Bucci[2], O. Semeraro[1], L. Cordier[3] & L. Mathelin[1]

1) LISN, CNRS, Université Paris-Saclay, Orsay (FR)
2) TAU, Inria, Université Paris-Saclay, CNRS, LISN, Orsay, (FR)
3) CNRS, Université de Poitiers, ISAE-ENSMA

Mail: michele-alessandro.bucci@inria.fr

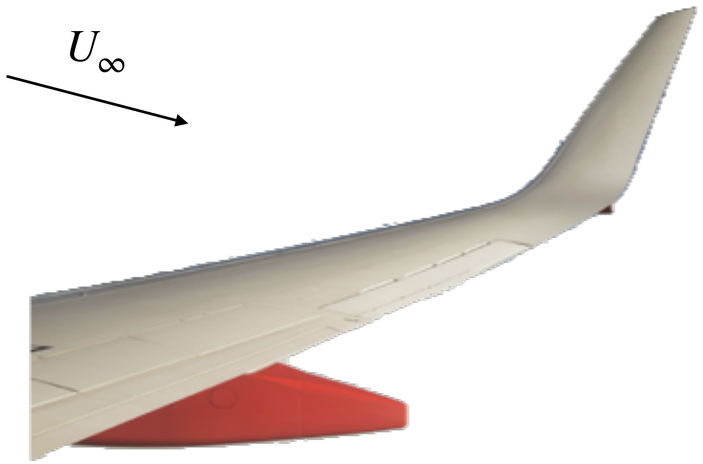
"Flowcon", ANR-17-ASTR-0022

Introduction

A complex (real) case: **flow over a wing**

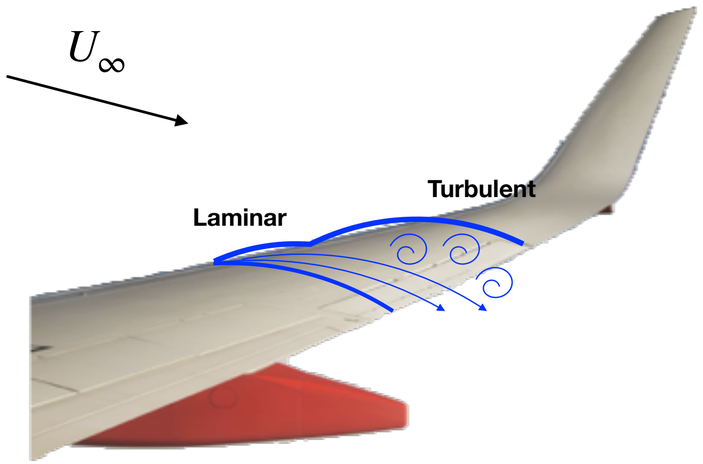
Introduction

A complex (real) case: **flow over a wing**



Introduction

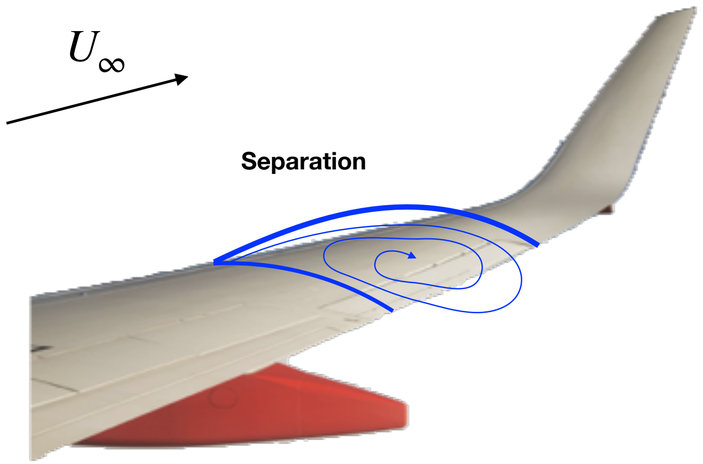
A complex (real) case: **flow over a wing**



- Laminar to turbulent transition

Introduction

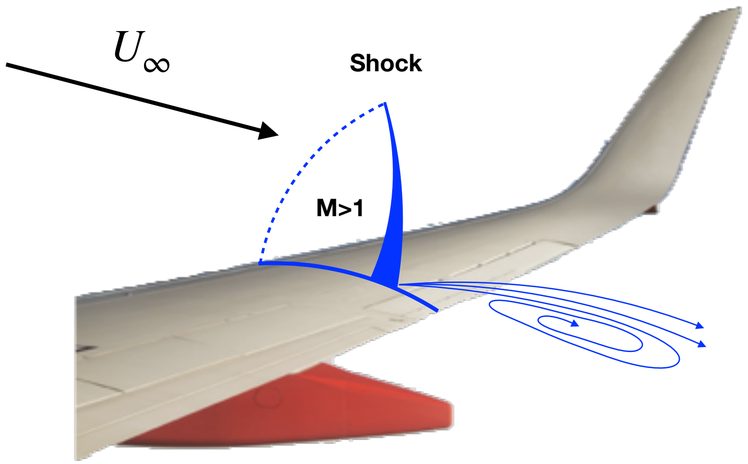
A complex (real) case: **flow over a wing**



- Laminar to turbulent transition
- Separation zone

Introduction

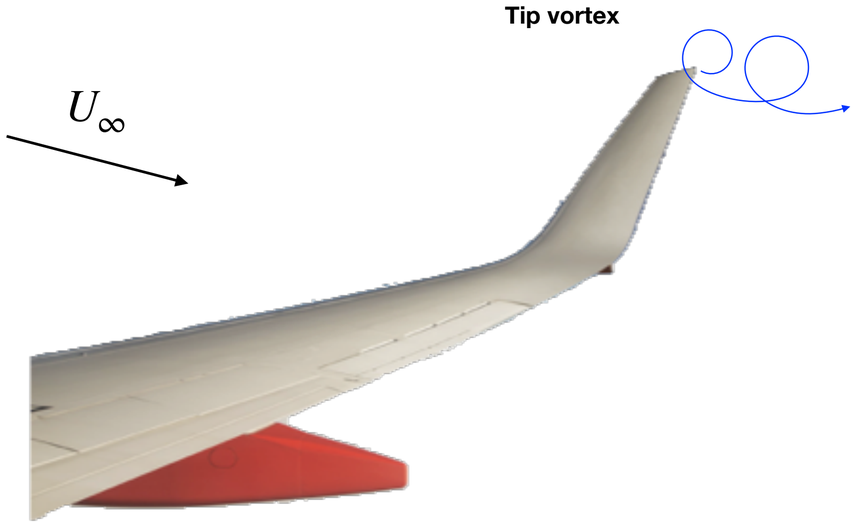
A complex (real) case: **flow over a wing**



- Laminar to turbulent transition
- Separation zone
- Buffet (λ)

Introduction

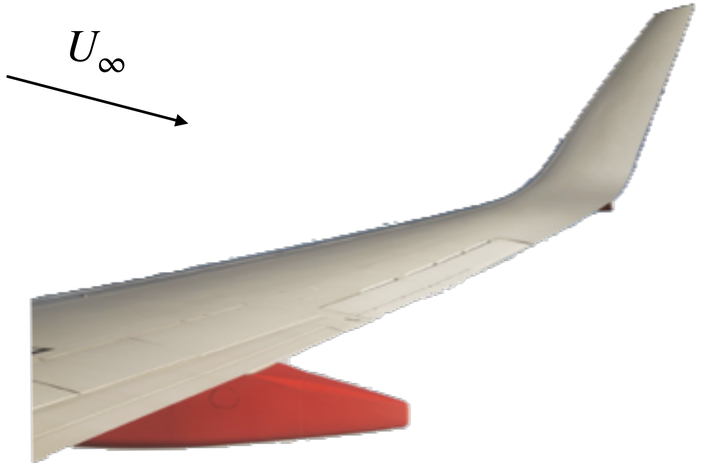
A complex (real) case: **flow over a wing**



- Laminar to turbulent transition
- Separation zone
- Buffet (λ)
- Secondary flows

Introduction

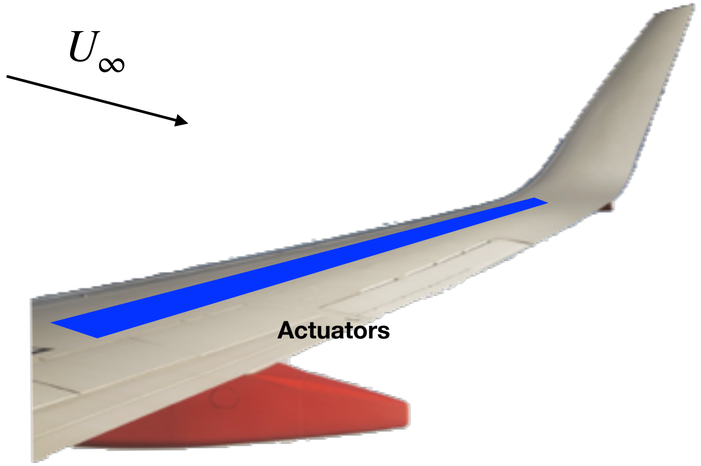
A complex (real) case: **flow over a wing**



- Laminar to turbulent transition
- Separation zone
- Buffet (λ)
- Secondary flows
- Fluid structure interaction
- ...

Introduction

A complex (real) case: **flow over a wing**



- Laminar to turbulent transition
- Separation zone
- Buffet (λ)
- Secondary flows
- Fluid structure interaction
- ...

Interact with the system **locally** to:

$$\mathcal{J} = \min\{\textbf{drag}\}, \max\{\textbf{lift}\}, \min\{\textbf{noise}\} \ldots$$

Linearized approach: the optimal case

Optimization problem

$$\mathcal{J} = \min_u \int_0^\infty (x^T Q x + u^T R u) dt$$

s.t. $\dot{x} = Ax + Bu$

$x \in \mathbb{R}^n$ sensors signal (state)

$u \in \mathbb{R}^m$ actuators signal (action)

Linearized approach: the optimal case

Optimization problem

$$\mathcal{J} = \min_u \int_0^\infty (x^T Q x + u^T R u) dt$$

s.t. $\dot{x} = Ax + Bu$

$x \in \mathbb{R}^n$ sensors signal (state)

$u \in \mathbb{R}^m$ actuators signal (action)

Augmented Lagrangian

$$\mathcal{J} = \underbrace{\min_u \int_0^\infty (x^T Q x + u^T R u) dt}_{\text{objective function}}$$

Linearized approach: the optimal case

Optimization problem

$$\mathcal{J} = \min_u \int_0^\infty (x^T Q x + u^T R u) dt \qquad x \in \mathbb{R}^n \quad \text{sensors signal (state)}$$

$$\text{s.t.} \quad \dot{x} = Ax + Bu \qquad u \in \mathbb{R}^m \quad \text{actuators signal (action)}$$

Augmented Lagrangian

$$\mathcal{J} = \min_u \underbrace{\int_0^\infty (x^T Q x + u^T R u) dt}_{\text{objective function}} - \lambda \underbrace{(\dot{x} - Ax - Bu)}_{\text{constraints}} \qquad \lambda \in \mathbb{R}^n \text{ co-state}$$

Linearized approach: the optimal case

Optimization problem

$$\mathcal{J} = \min_u \int_0^\infty (x^T Q x + u^T R u) dt \qquad x \in \mathbb{R}^n \quad \text{sensors signal (state)}$$

$$\text{s.t.} \quad \dot{x} = Ax + Bu \qquad u \in \mathbb{R}^m \quad \text{actuators signal (action)}$$

Augmented Lagrangian

$$\mathcal{J} = \min_u \underbrace{\int_0^\infty (x^T Q x + u^T R u) dt}_{\text{objective function}} - \lambda \underbrace{(\dot{x} - Ax - Bu)}_{\text{constraints}} \qquad \lambda \in \mathbb{R}^n \text{ co-state}$$

$$\frac{\partial \mathcal{J}}{\partial \lambda} = 0 \Rightarrow \dot{x} = \left(\frac{\partial H}{\partial \lambda}\right)^T = Ax + Bu \quad x(0) = x_0$$

$$\frac{\partial \mathcal{J}}{\partial x} = 0 \Rightarrow -\dot{\lambda} = \left(\frac{\partial H}{\partial x}\right)^T = Qx + A^T \lambda \quad \lambda(T) = 0$$

$$\frac{\partial \mathcal{J}}{\partial u} = 0 \Rightarrow 0 = \left(\frac{\partial H}{\partial u}\right)^T = Ru + \lambda^T B = 0$$

Linear methods

$$\frac{\partial \mathcal{J}}{\partial \lambda} = 0 \Rightarrow \dot{x} = \left(\frac{\partial H}{\partial \lambda}\right)^T = Ax + Bu \quad x(0) = x_0$$

$$\frac{\partial \mathcal{J}}{\partial x} = 0 \Rightarrow -\dot{\lambda} = \left(\frac{\partial H}{\partial x}\right)^T = Qx + A^T\lambda \quad \lambda(T) = 0$$

$$\frac{\partial \mathcal{J}}{\partial u} = 0 \Rightarrow 0 = \left(\frac{\partial H}{\partial x}\right)^T = Ru + \lambda^T B = 0$$

Linear methods

$$\frac{\partial \mathcal{J}}{\partial \lambda} = 0 \Rightarrow \dot{x} = \left(\frac{\partial H}{\partial \lambda}\right)^T = Ax + Bu \quad x(0) = x_0$$

$$\frac{\partial \mathcal{J}}{\partial x} = 0 \Rightarrow -\dot{\lambda} = \left(\frac{\partial H}{\partial x}\right)^T = Qx + A^T\lambda \quad \lambda(T) = 0$$

$$\frac{\partial \mathcal{J}}{\partial u} = 0 \Rightarrow 0 = \left(\frac{\partial H}{\partial x}\right)^T = Ru + \lambda^T B = 0$$

Iterative methods: Direct-Adjoint
Linear iteration, Adjoint of direct adjoint (ADA), extension to nonlinear cases, Model Predictive Control (MPC)…

Direct methods: Control Algebraic **Riccati Equation**

Kim, J. and Bewley T.R. (2007). Annual Review of Fluid Mech.
Luchini, P. and Bottaro, A. (2014). Annual Review of Fluid Mech.

Optimal control: the ***non***linear case

$$\mathcal{J}(x(t), t) = \max_u \int_t^\infty (x^T Q x + u^T R u) dt \quad st \qquad \dot{x} = Ax + Bu$$

- The integrand of the cost function is the **reward** $r(x, u)$
- **Non linear** dynamical system
- the objective is to find the **policy** $u = \pi(x)$ that maximise the **value function**

Optimal control: the ***non***linear case

$$\mathcal{J}(x(t), t) = \max_u \int_t^\infty r(x, u)dt \qquad st \qquad \dot{x} = f(x, u)$$

- The integrand of the cost function is the **reward** $r(x, u)$
- **Non linear** dynamical system
- the objective is to find the **policy** $u = \pi(x)$ that maximise the **value function**

Hamilton-Jacobi-Bellman (HJB) equation

$$\mathcal{J}(x(t), t) = \max_u \int_t^{t+\Delta t} r(x, u)dt + \underbrace{\mathcal{J}(x(t + \Delta t), t + \Delta t)}_{\text{future value function}}$$

The principle of optimality requires that the **future value function** has to be **maximised** expanding in series and taking $\Delta t \to 0$

Hamilton-Jacobi-Bellman (HJB) equation

$$\mathcal{J}(x(t),t) = \max_u \int_t^{t+\Delta t} r(x,u)dt + \underbrace{\mathcal{J}(x(t+\Delta t), t+\Delta t)}_{\text{future value function}}$$

The principle of optimality requires that the **future value function** has to be **maximised** expanding in series and taking $\Delta t \to 0$

$$-\mathcal{J}(\dot{x(t)},t) = \max_u \{ r(x,u) + \nabla \mathcal{J}(x(t),t) f(x,u) \}$$

The **Hamilton-Jacobi-Bellman** eq. (1954) is a functional equation.

The solution is the **optimal policy** $u = \pi(x)$

HJB in discrete form

Discounted HJB

We introduce in the HJB the term $\gamma = e^{-\rho t}$ where $\rho > 0$ is the **discount factor**

$$\mathcal{J}(x(t), t) = \max_u \int_t^T e^{-\rho t} r(x, u) dt$$

$$\rho \mathcal{J}(x(t), t) = \max_u \{ r(x, u) + \nabla \mathcal{J}(x(t), t) f(x, u) \}$$

Discrete time observations

Bellman optimality equation $\mathcal{J}(x_n) = \max_u \{ r(x_n, u) + \gamma \mathcal{J}(x_{n+1}) \}$

Bellman equation

Bellman optimality equation $\mathcal{J}(x_n) = \max_u \{ r(x_n, u) + \gamma \mathcal{J}(x_{n+1}) \}$

Bellman equation

Bellman optimality equation $\mathcal{J}(x_n) = \max_u \{ r(x_n, u) + \gamma \mathcal{J}(x_{n+1}) \}$

Evolution of the discounted cost function following an optimal policy

$$V_\pi(x_n) = r(x_n, u_n) + \gamma V_\pi(x_{n+1})$$ *Value* function

$$Q_\pi(x_n, u_n) = r(x_n, u_n) + \gamma Q_\pi(x_{n+1}, u_{n+1})$$ *Quality* function

Bellman equation

Bellman optimality equation $\mathcal{J}(x_n) = \max_u \{ r(x_n, u) + \gamma \mathcal{J}(x_{n+1}) \}$

Evolution of the discounted cost function following an optimal policy

$$V_\pi(x_n) = r(x_n, u_n) + \gamma V_\pi(x_{n+1})$$ *Value* function

$$Q_\pi(x_n, u_n) = r(x_n, u_n) + \gamma Q_\pi(x_{n+1}, u_{n+1})$$ *Quality* function

Bellman's principle of optimality

The discounted infinite-horizon optimal problem can be decomposed in a series of local optimal problems.

Bellman equation

Bellman optimality equation $\mathcal{J}(x_n) = \max_u \{ r(x_n, u) + \gamma \mathcal{J}(x_{n+1}) \}$

Evolution of the discounted cost function following an optimal policy

$$V_\pi(x_n) = r(x_n, u_n) + \gamma V_\pi(x_{n+1})$$ *Value* function

$$Q_\pi(x_n, u_n) = r(x_n, u_n) + \gamma Q_\pi(x_{n+1}, u_{n+1})$$ *Quality* function

Bellman's principle of optimality

The discounted infinite-horizon optimal problem can be decomposed in a series of local optimal problems.

Reinforcement Learning

It is possible to **learn** from the **observation** of the interaction between the system and the plant

Case study

Control of Kuramoto-Sivashinsky system

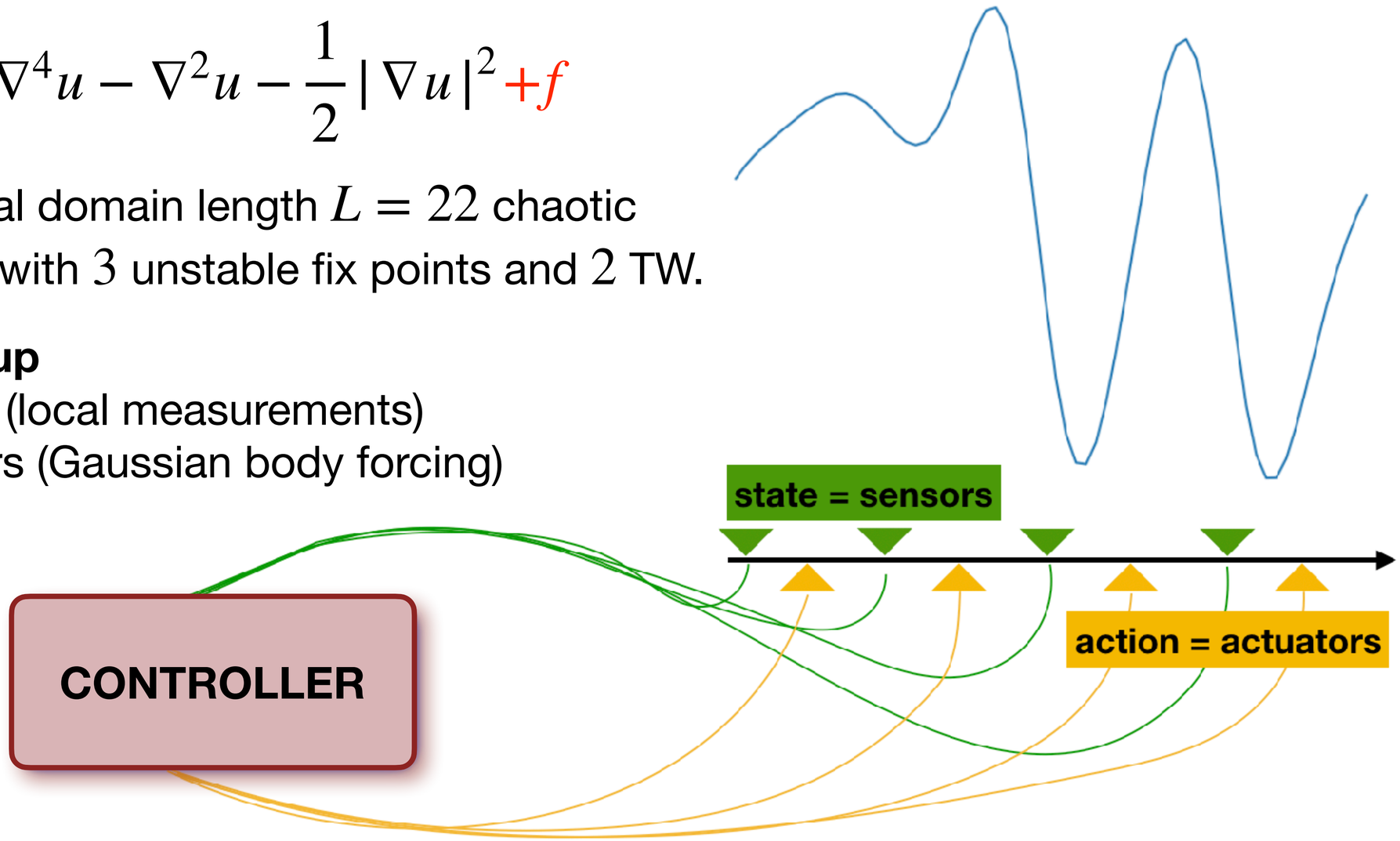
Bucci M. A., Semeraro O., Allauzen A., Wisniewski G., Cordier L. & Mathelin L.. "Control of chaotic systems by deep reinforcement learning." *Proceedings of the Royal Society A* (2019)

KS system as a plant

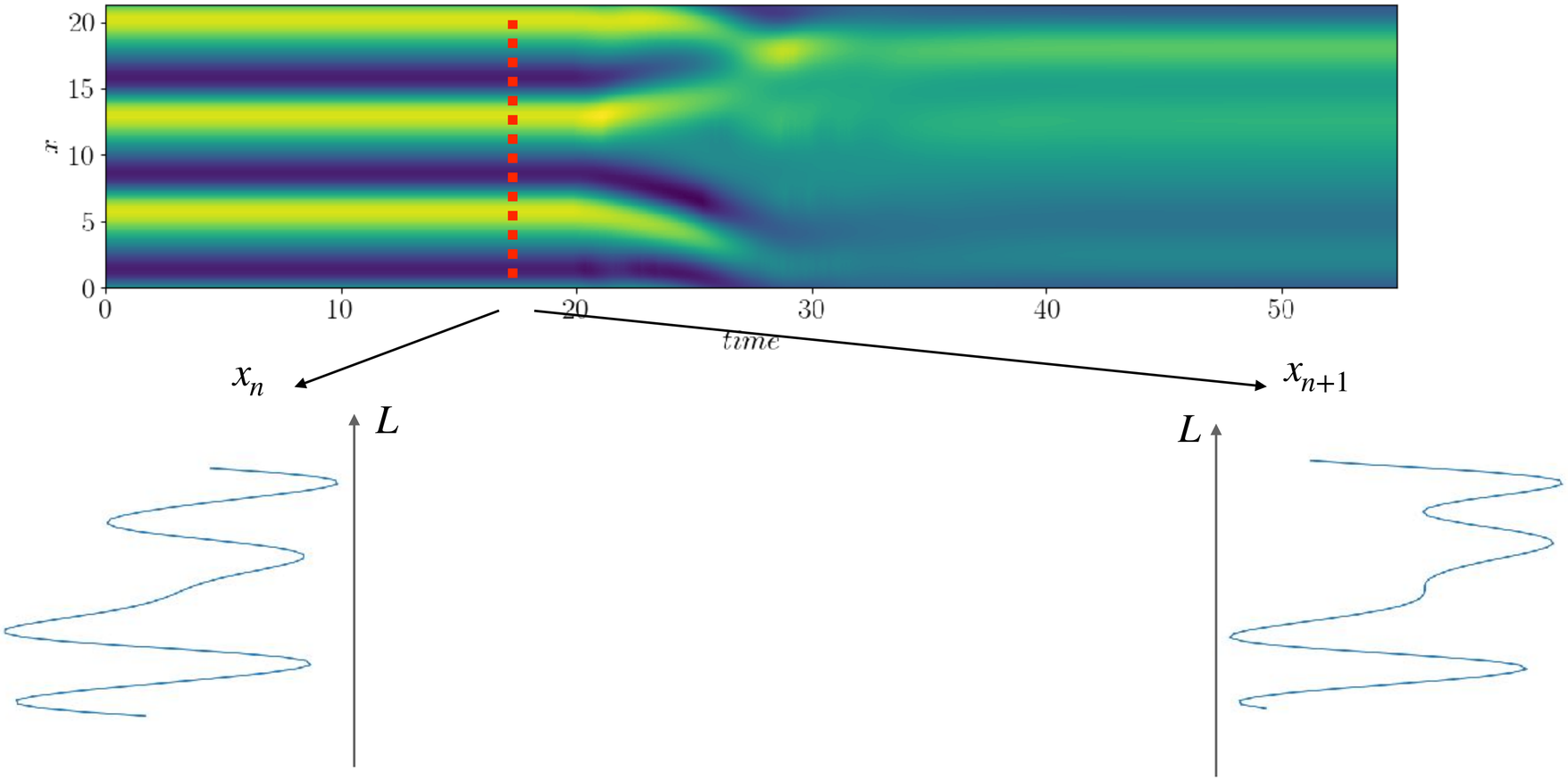
$$\frac{\partial u}{\partial t} = -\nabla^4 u - \nabla^2 u - \frac{1}{2}|\nabla u|^2 + f$$

If numerical domain length $L = 22$ chaotic dynamics with 3 unstable fix points and 2 TW.

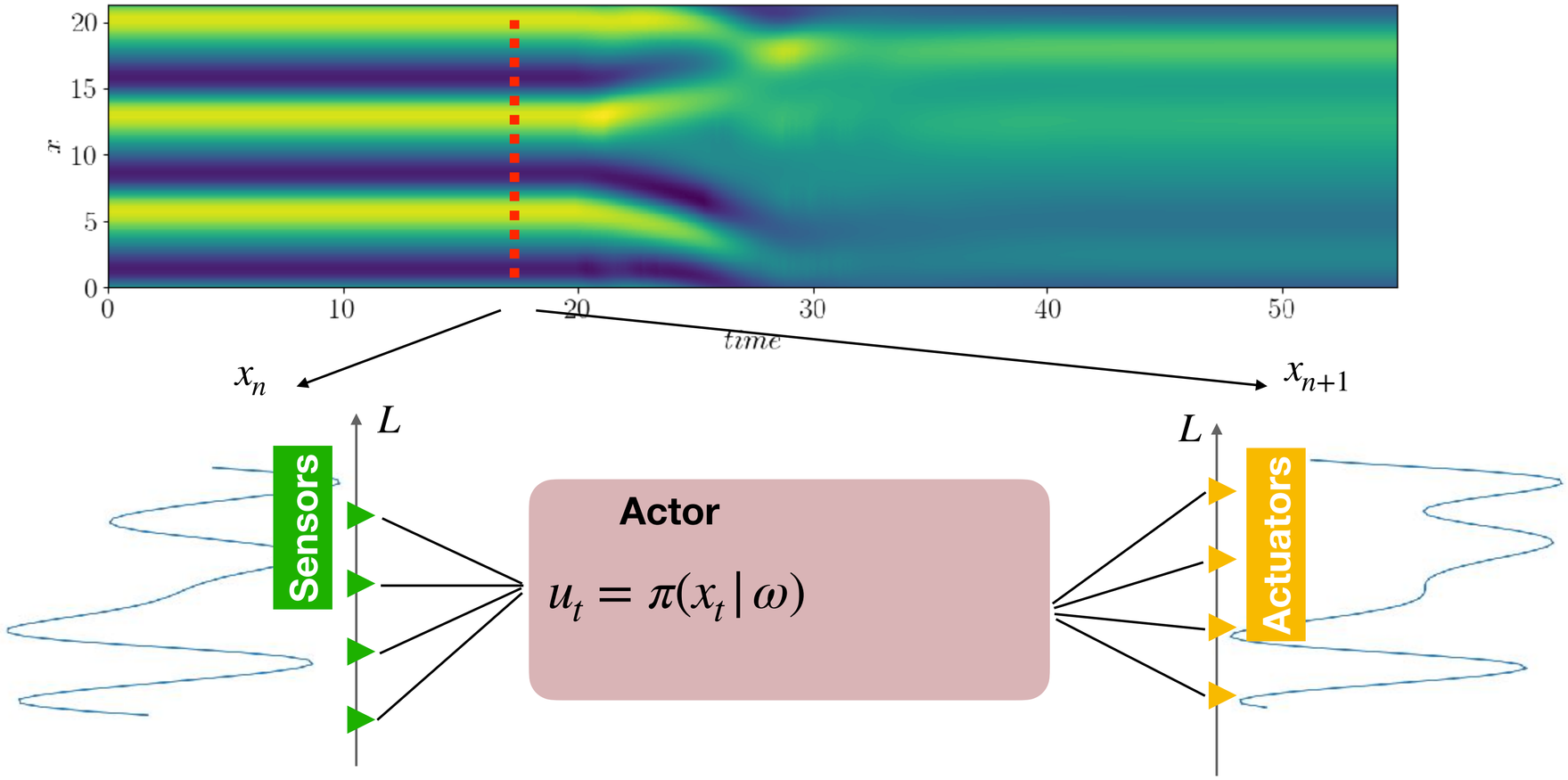
Plant setup
8 sensors (local measurements)
4 actuators (Gaussian body forcing)



RL



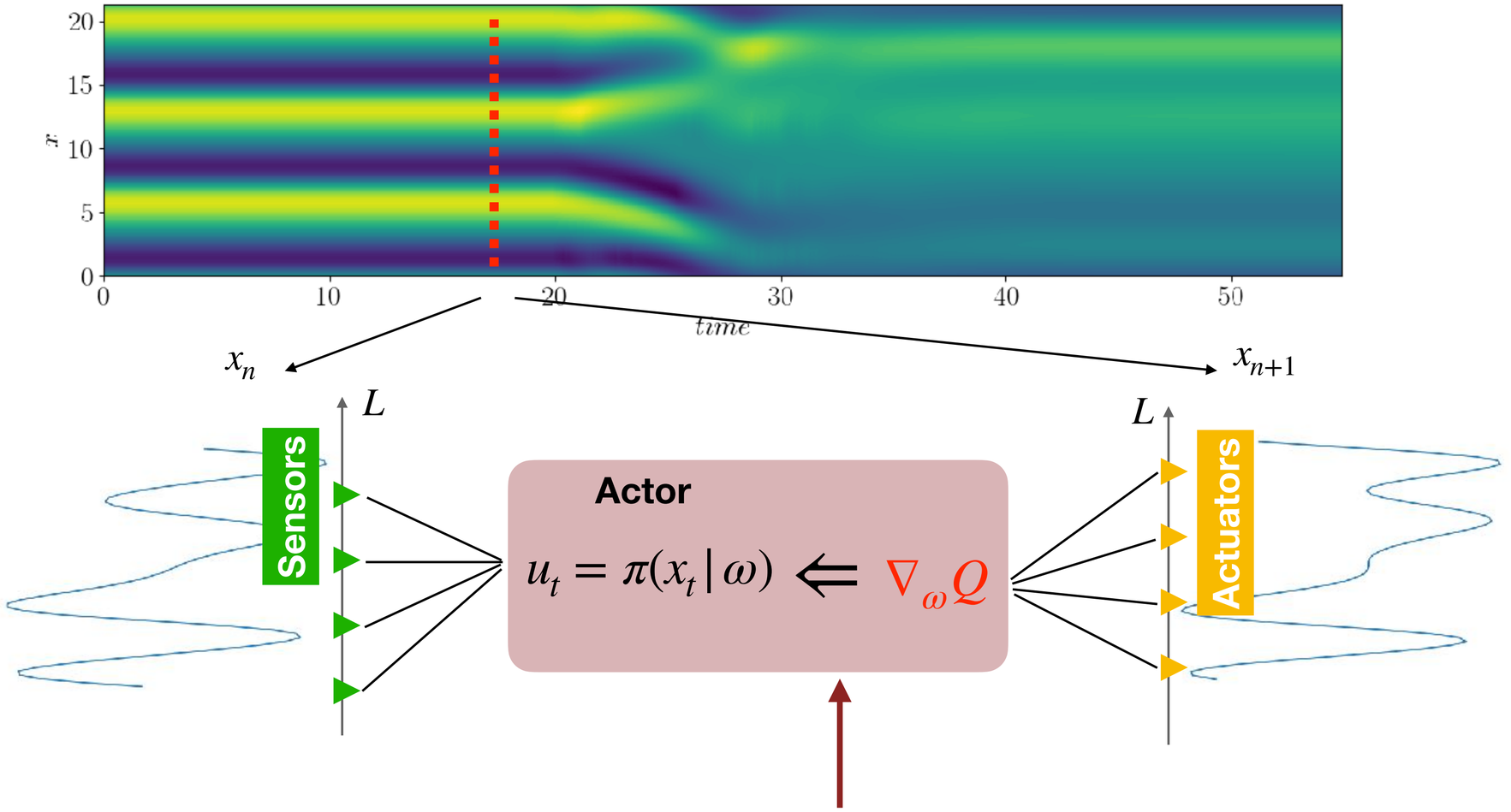
RL



Critic

$Q(x_t, u_t \mid \theta)$

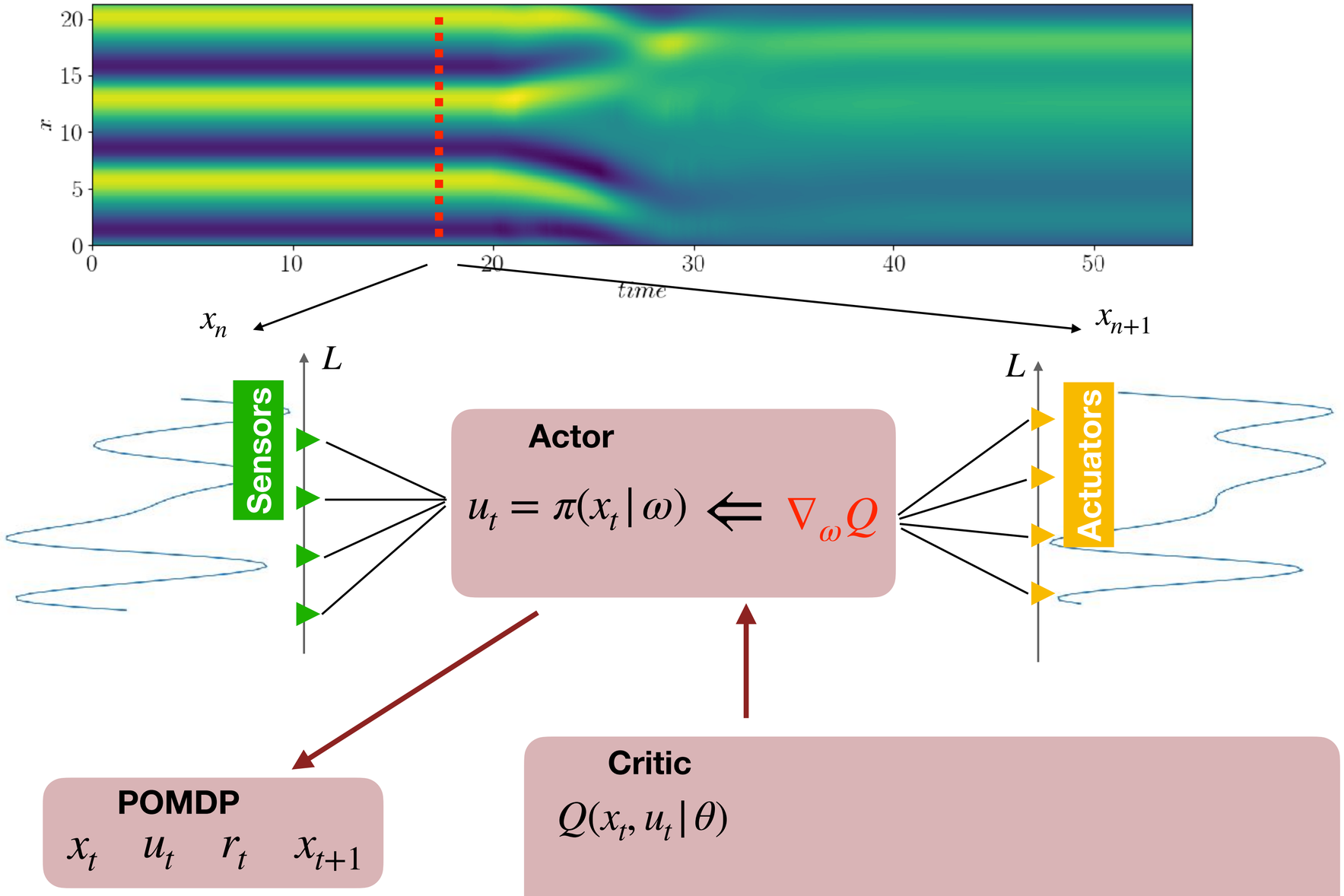
RL



Critic

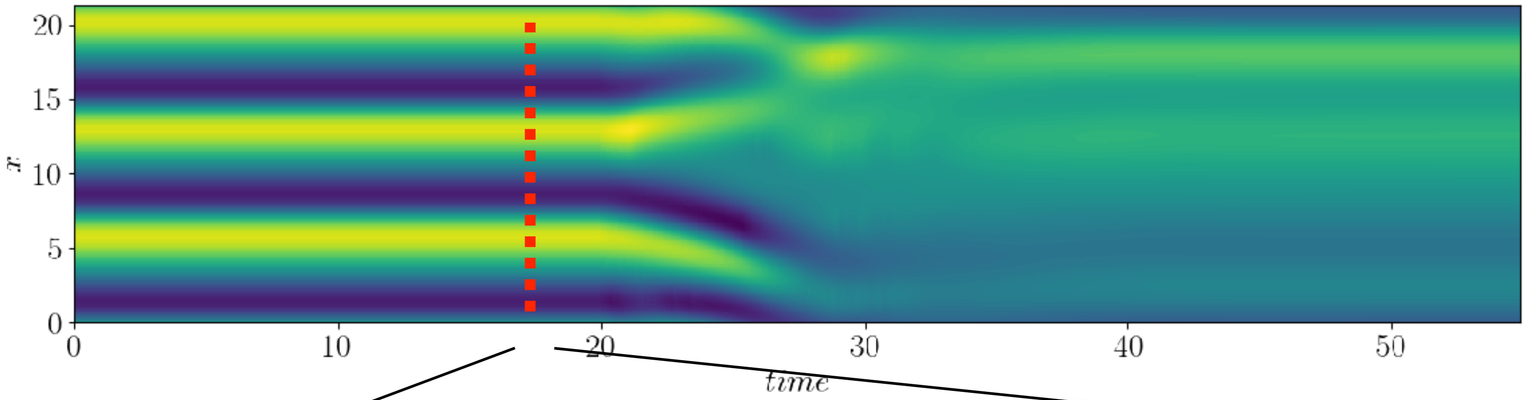
$Q(x_t, u_t | \theta)$

RL

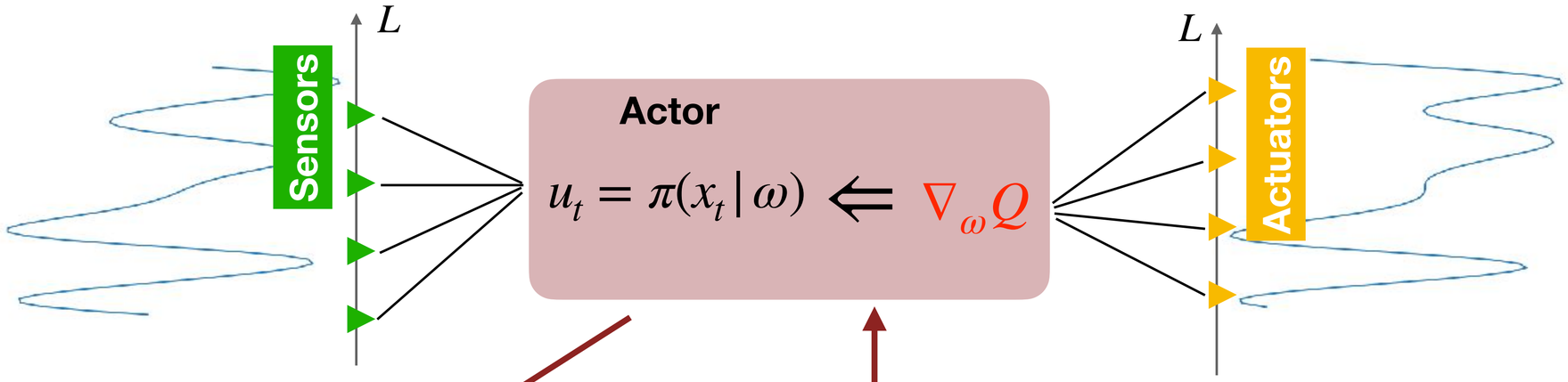


PODMD: partial observable Markov Decision Proces

RL



x_n x_{n+1}



Actor

$$u_t = \pi(x_t \,|\, \omega) \Leftarrow \nabla_\omega Q$$

POMDP

$x_t \quad u_t \quad r_t \quad x_{t+1}$

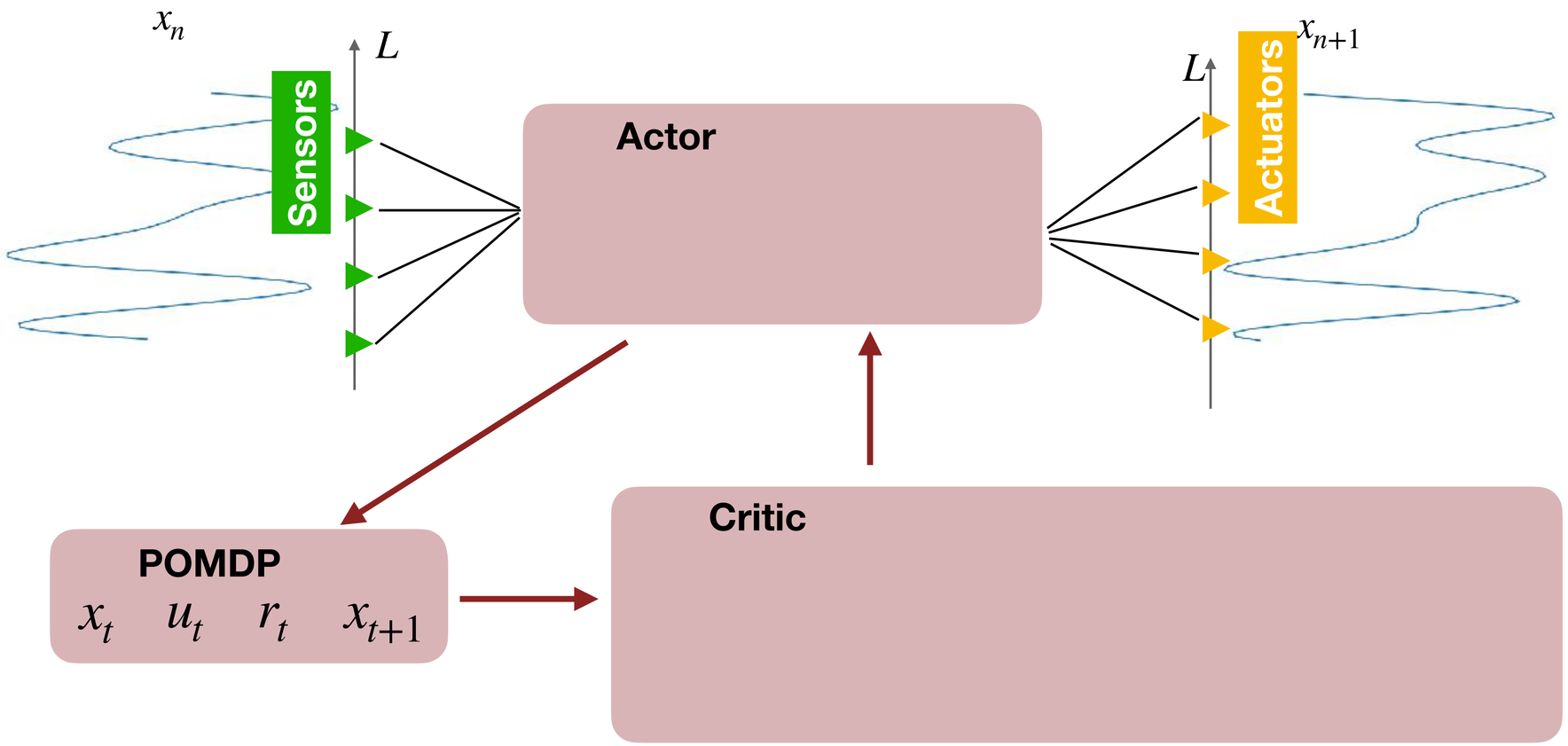
Critic

$$Q(x_t, u_t \,|\, \theta) \Leftarrow \nabla_\theta TD$$

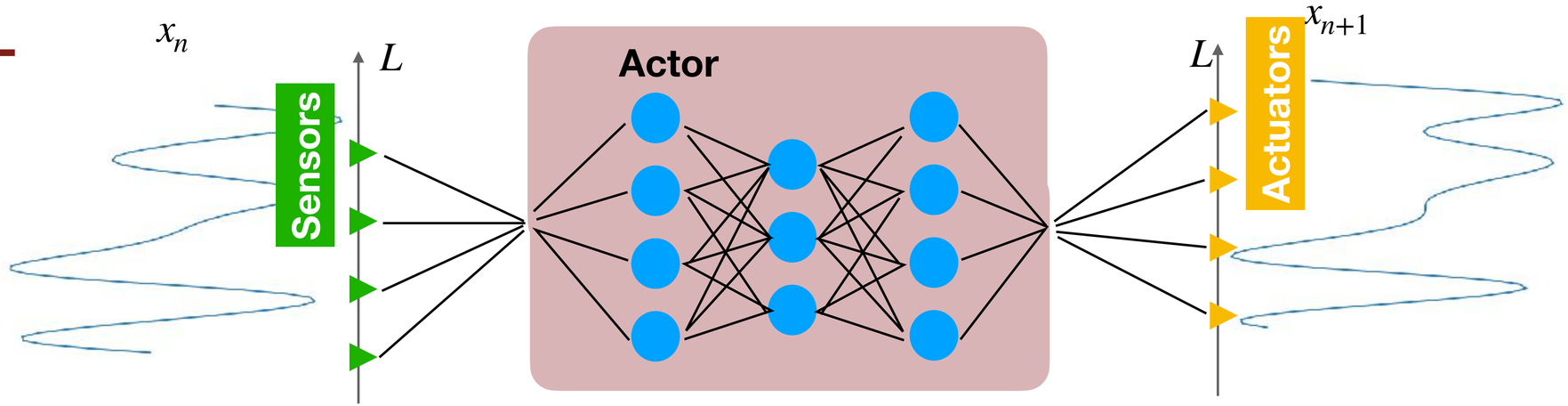
$$TD := Q(x_t, u_t \,|\, \theta) - r_t - \gamma Q(x_{t+1}, \pi(x_{t+1}) \,|\, \theta)$$

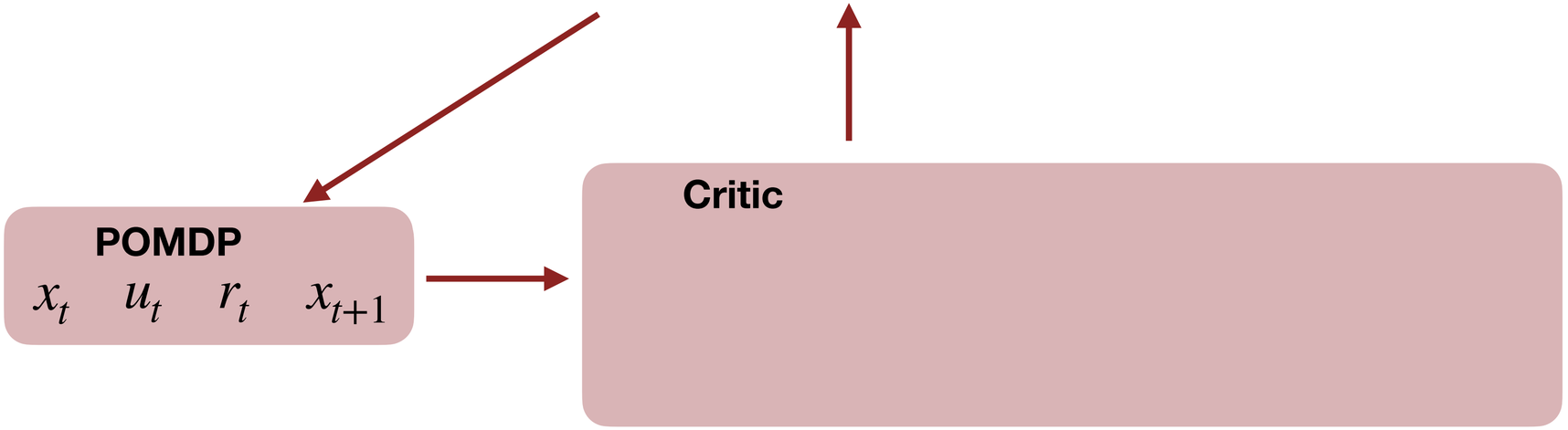
PODMD: partial observable Markov Decision Proces

Deep RL

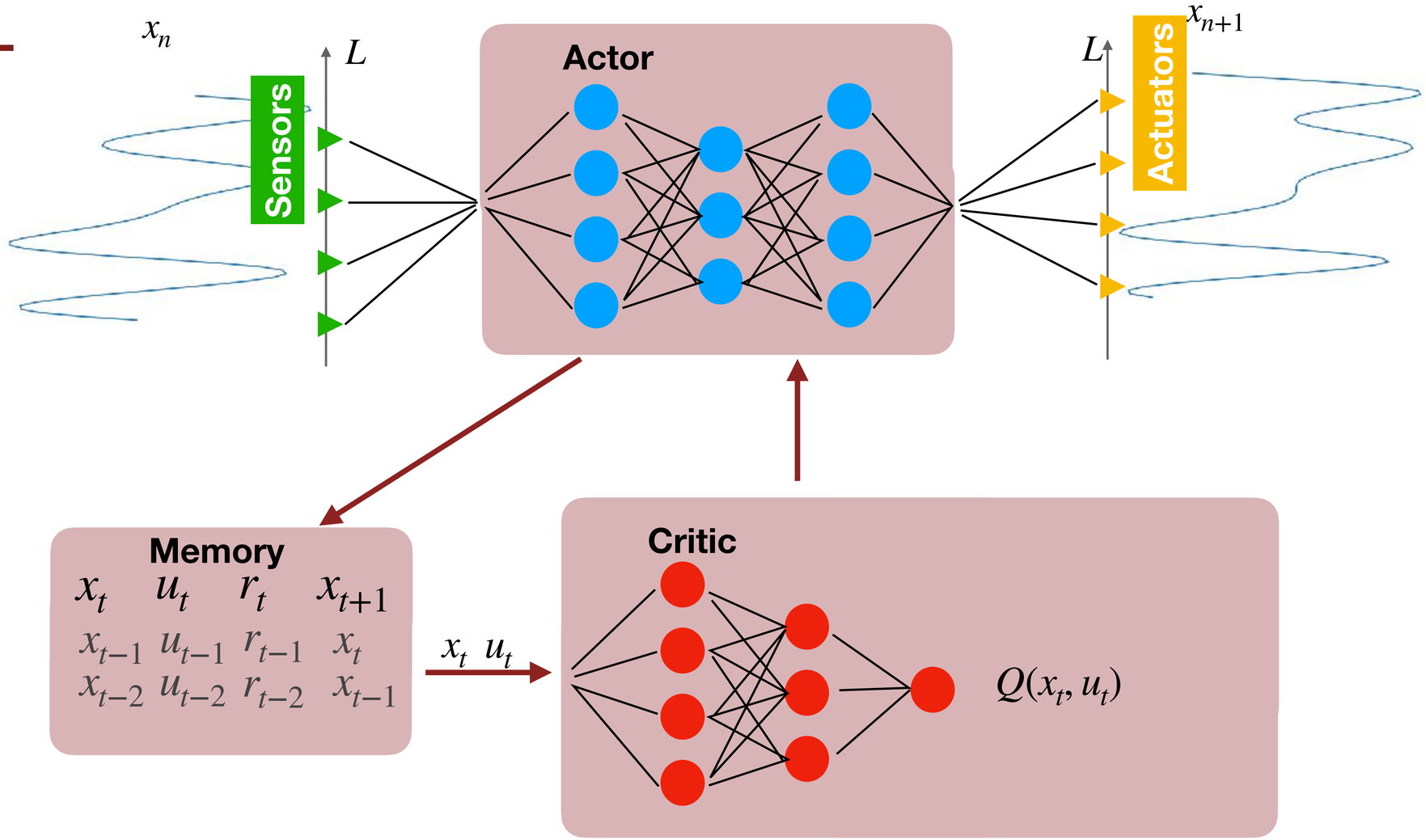


Deep RL

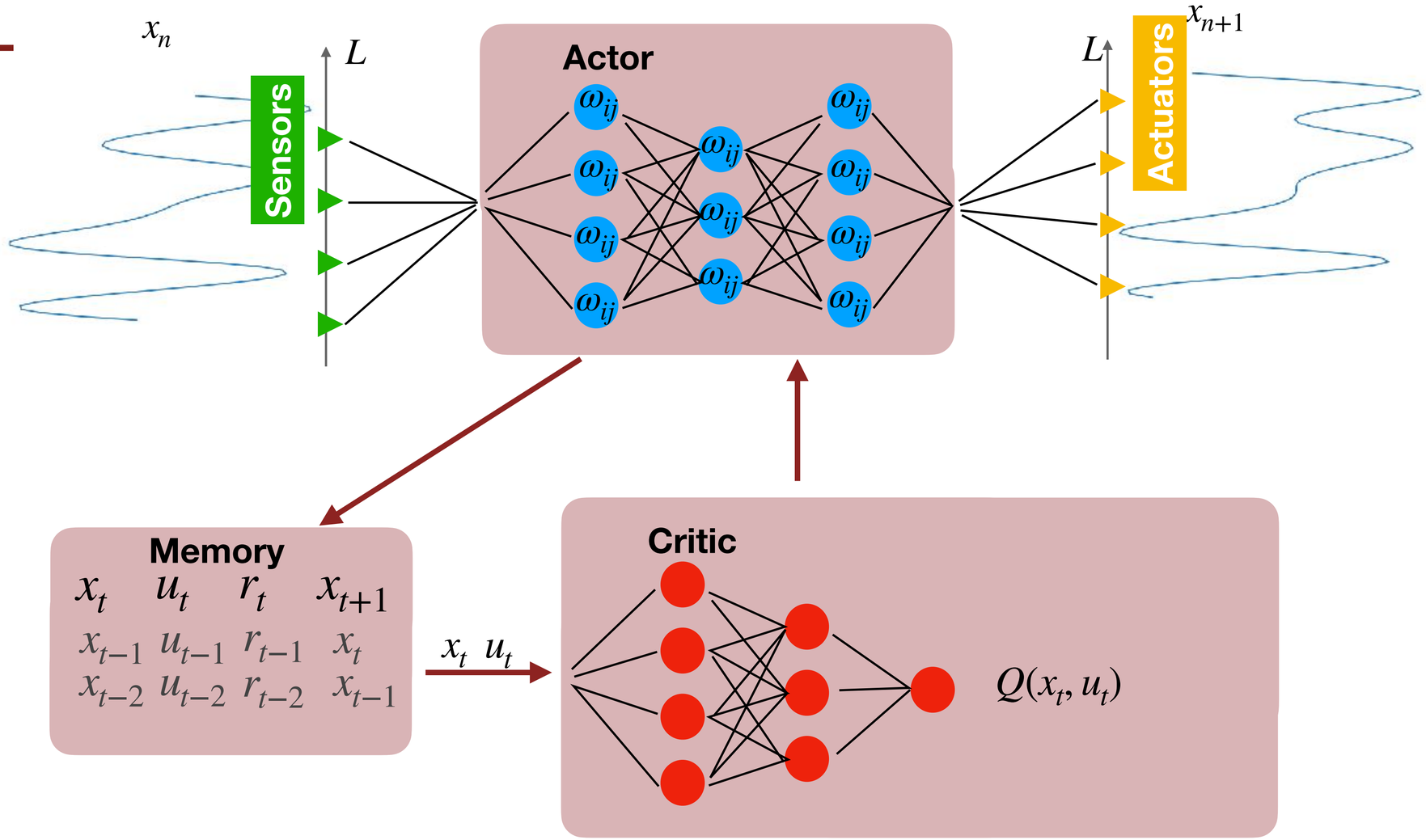




Deep RL

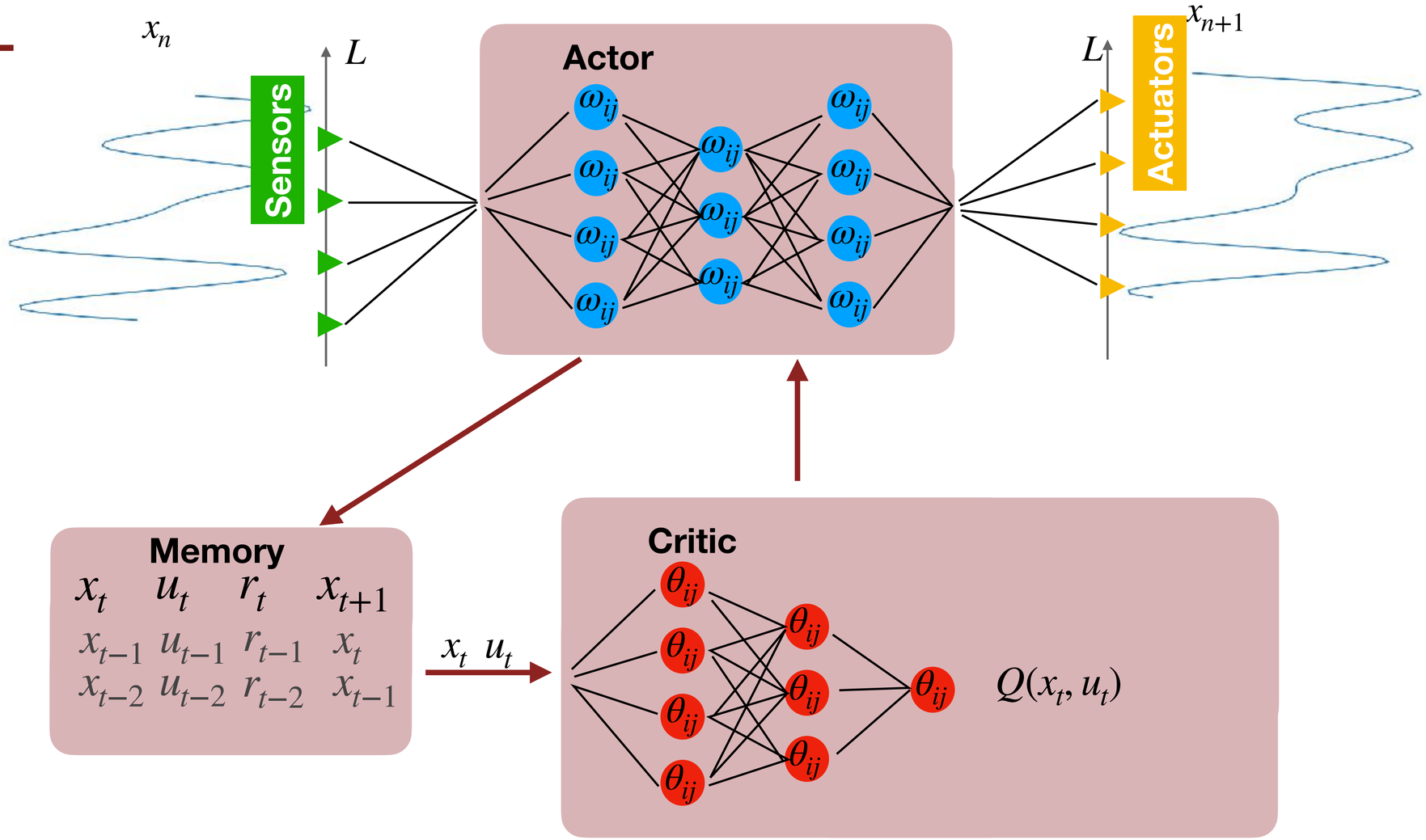

x_n
L
Sensors
Actor
L
Actuators
x_{n+1}
Memory
x_t u_t r_t x_{t+1}
x_{t-1} u_{t-1} r_{t-1} x_t
x_{t-2} u_{t-2} r_{t-2} x_{t-1}
x_t u_t
Critic
$Q(x_t, u_t)$

Deep RL



ω_{ij} are the weights of the **Actor Neural Network**

Deep RL



ω_{ij} are the weights of the **Actor Neural Network**

θ_{ij} are the weights of the **Critic Neural Network**

Deep Deterministic Policy Gradient

Critic update

$$\nabla_\theta TD = \frac{\partial \| Q_t - (r_t + Q_{t+1}) \|}{\partial \theta_i}$$

$$\theta_i' = \theta_i - \alpha \frac{\partial TD}{\partial \theta_i}$$

256 Neurons
3 Layers

Actor update

$$\nabla_\omega Q = \frac{\partial Q_t}{\partial u_t} \frac{\partial \pi_t}{\partial \omega_i}$$

$$\omega_i' = \omega_i + \alpha \frac{\partial Q}{\partial \omega_i}$$

128 Neurons
3 Layers

Learn from the observation → **Explore the action-state space**

Perturbation of the parameters of the policy, Ornstein-Uhlenbeck process

Deep Deterministic Policy Gradient

Critic update

$$\nabla_\theta TD = \frac{\partial \| Q_t - (r_t + Q_{t+1}) \|}{\partial \theta_i} \qquad \theta_i' = \theta_i - \alpha \frac{\partial TD}{\partial \theta_i}$$

256 Neurons
3 Layers

Actor update

$$\nabla_\omega Q = \frac{\partial Q_t}{\partial u_t} \frac{\partial \pi_t}{\partial \omega_i} \qquad \omega_i' = \omega_i + \alpha \frac{\partial Q}{\partial \omega_i}$$

128 Neurons
3 Layers

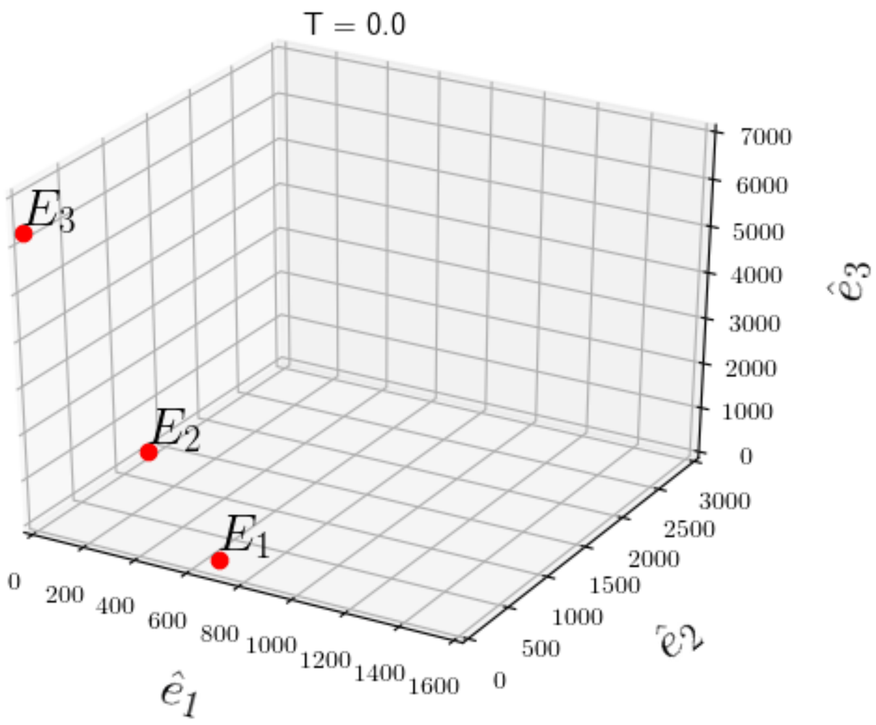
Learn from the observation → **Explore the action-state space**

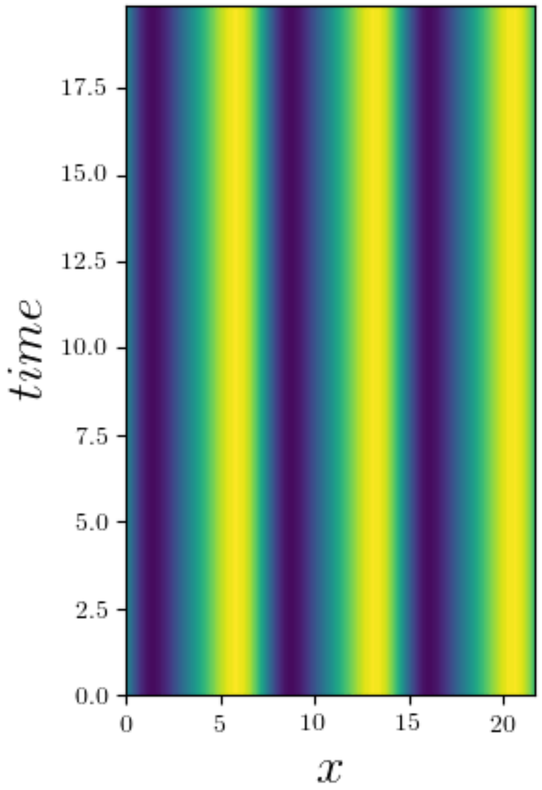
Perturbation of the parameters of the policy, Ornstein-Uhlenbeck process

Remark

For the Critic update we don't need to know the policy used to generate a_t:
off-policy approach

Controlled KS system





Actor-Critic algorithm: **deep deterministic policy gradient** (DDPG)

Three policies are identified

Reward defined with respect of the target states

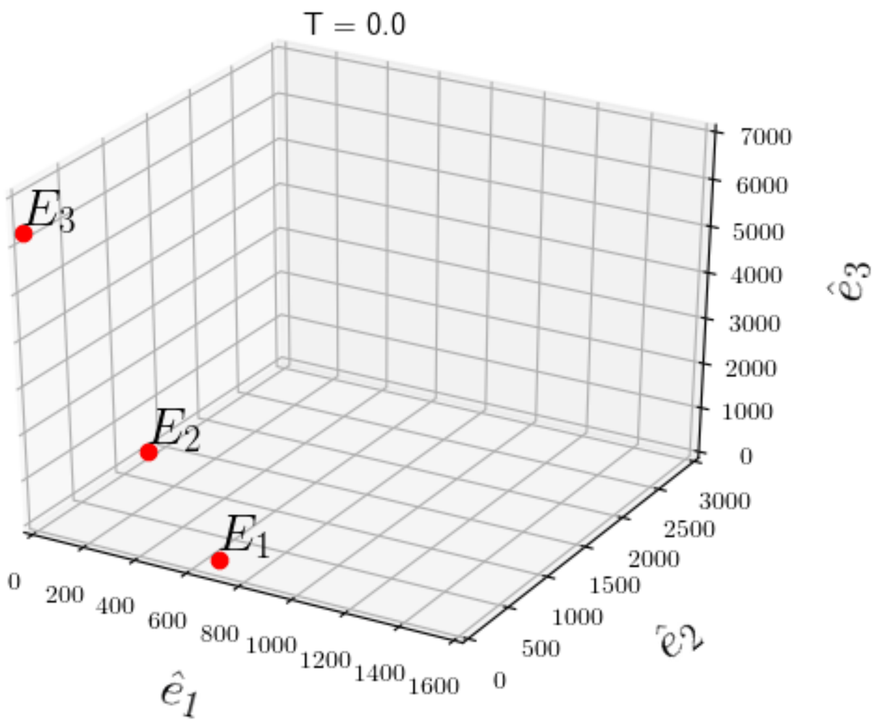
$$r_t = -\|x_t - x_{Target}\|$$

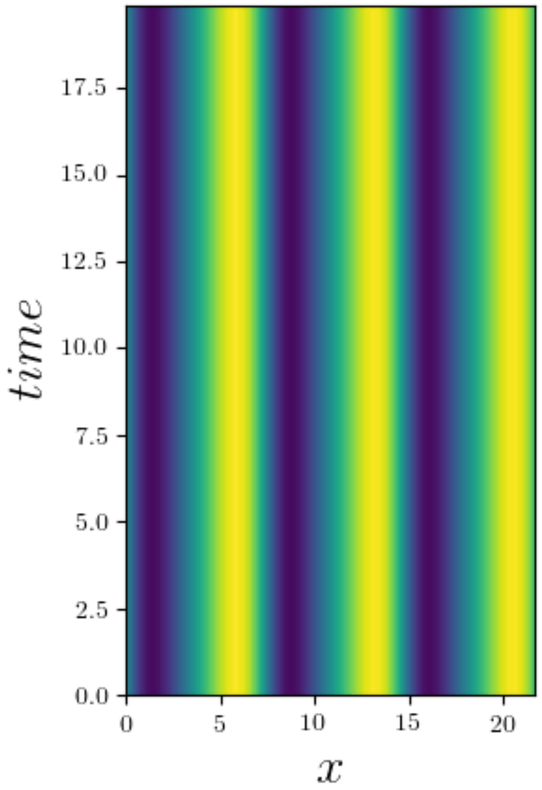
Discount factor

On average: **1 hour training** (Intel I3 CPU, 2015)

Silver, D., et al. (2014), ICML
Bucci, M.A. et al. (2019), PRSA

Controlled KS system





Actor-Critic algorithm: **deep deterministic policy gradient** (DDPG)

Three policies are identified

Reward defined with respect of the target states

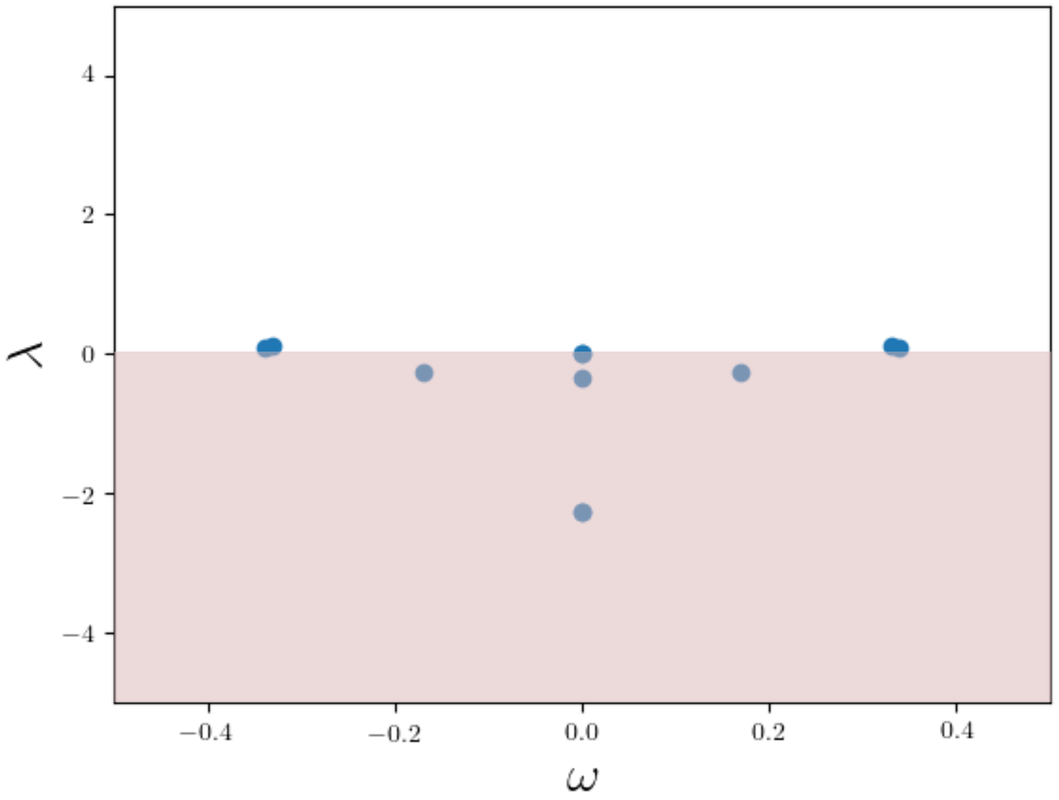
$$r_t = -\|x_t - x_{Target}\|$$

Discount factor

On average: **1 hour training** (Intel I3 CPU, 2015)

Silver, D., et al. (2014), ICML
Bucci, M.A. et al. (2019), PRSA

Controlled KS system



modal solution ansatz around

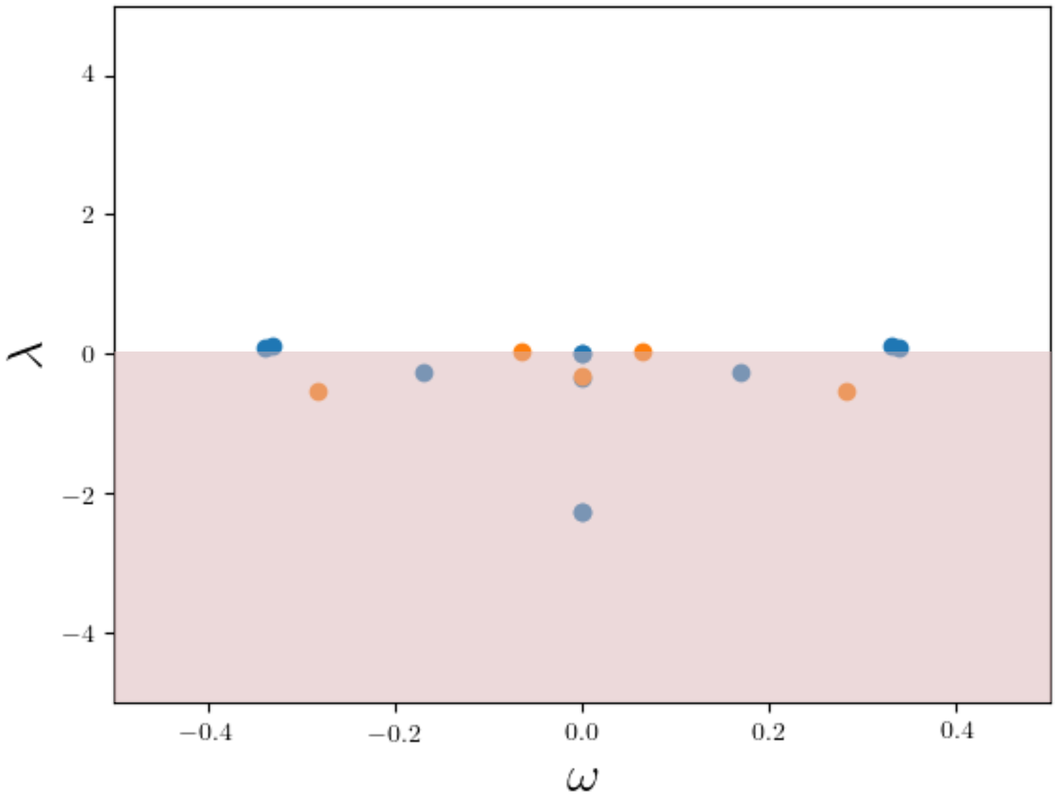
$$x = \hat{x} e^{(\lambda + i\omega)t}$$

Uncontrolled KS:

$$(\lambda + i\omega)\hat{x} = J_{KS}\hat{x}$$

two unstable eigenvalues

Controlled KS system



modal solution ansatz around

$$x = \hat{x} e^{(\lambda + i\omega)t}$$

Uncontrolled KS:

$$(\lambda + i\omega)\hat{x} = J_{KS}\hat{x}$$

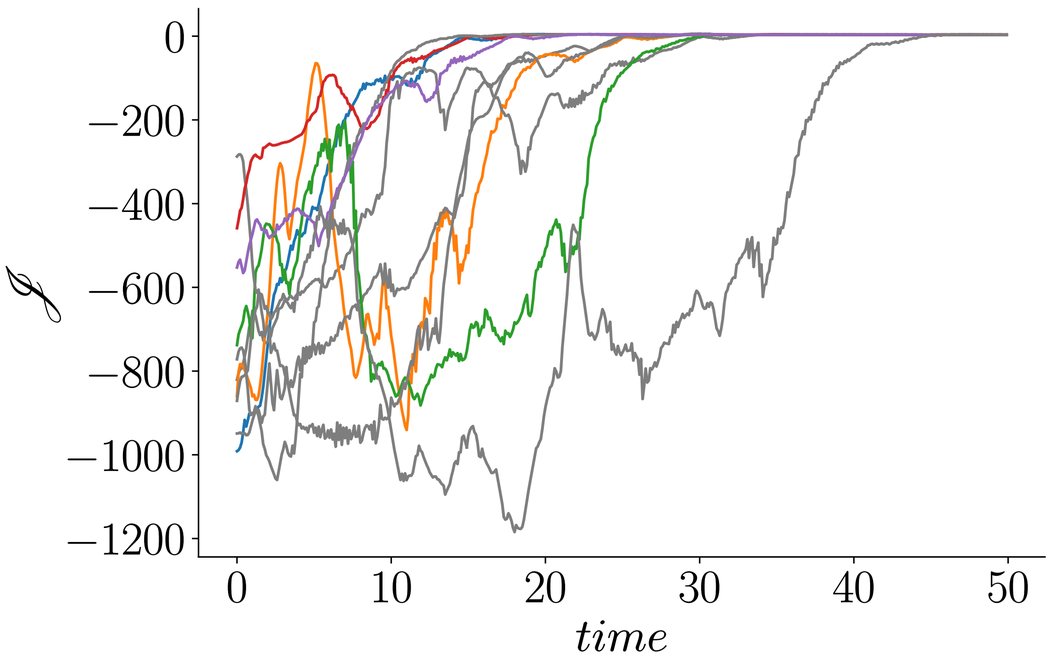
two unstable eigenvalues

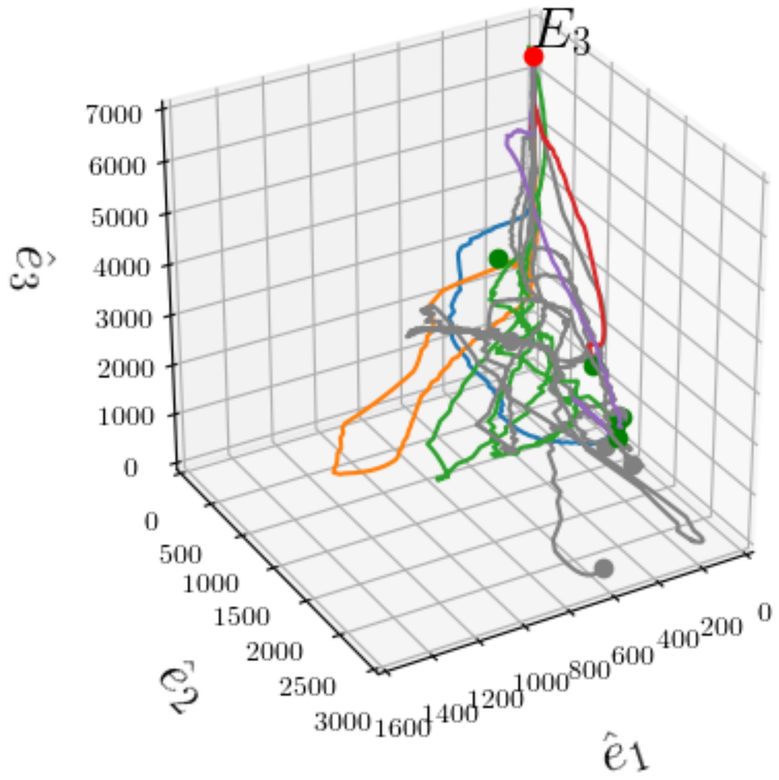
Controlled KS:

$$(\lambda + i\omega)\hat{x} = (J_{KS} + \pi_x)\hat{x}$$

no unstable eigenvalues

Controlled KS system





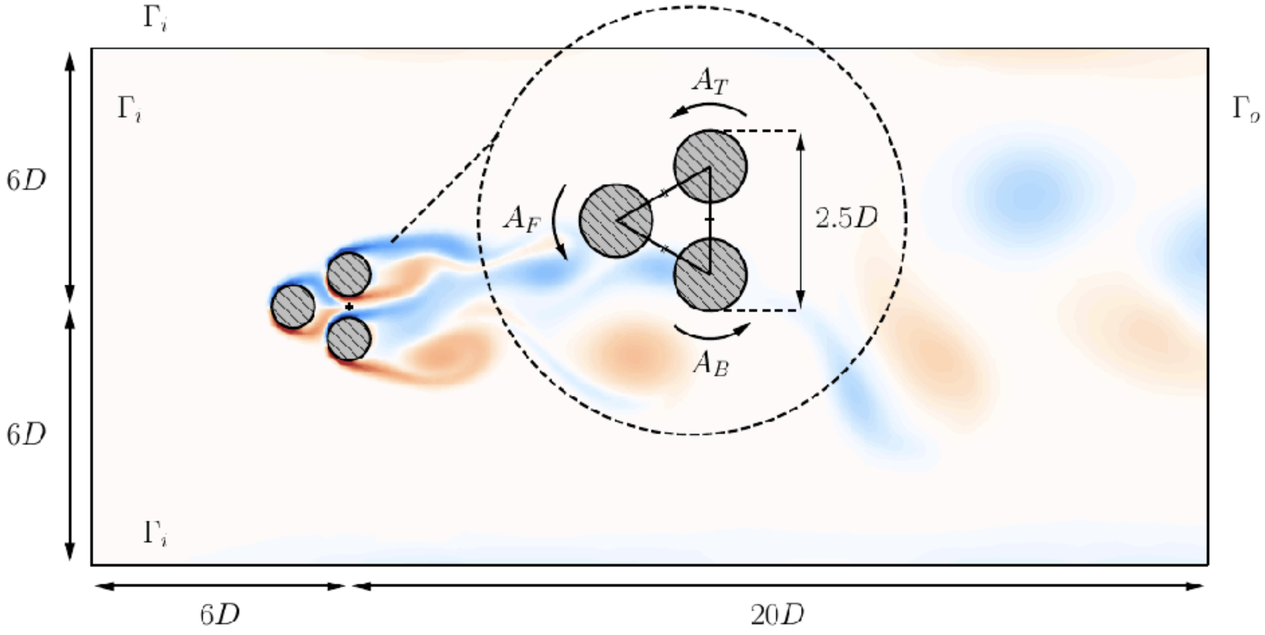
The identified policies are **robust** with respect of the initial conditions due to the **Markiovanity** of the **Q-function**, and the **exploration**

Case study

The fluidic Pinball

Guegan, T., Bucci, M. A., Semeraro, O., Cordier, L., & Mathelin, L. (2020). Control by Deep Reinforcement Learning of a separated flow. *Bulletin of the American Physical Society*.

The fluidic pinball case (CFD)



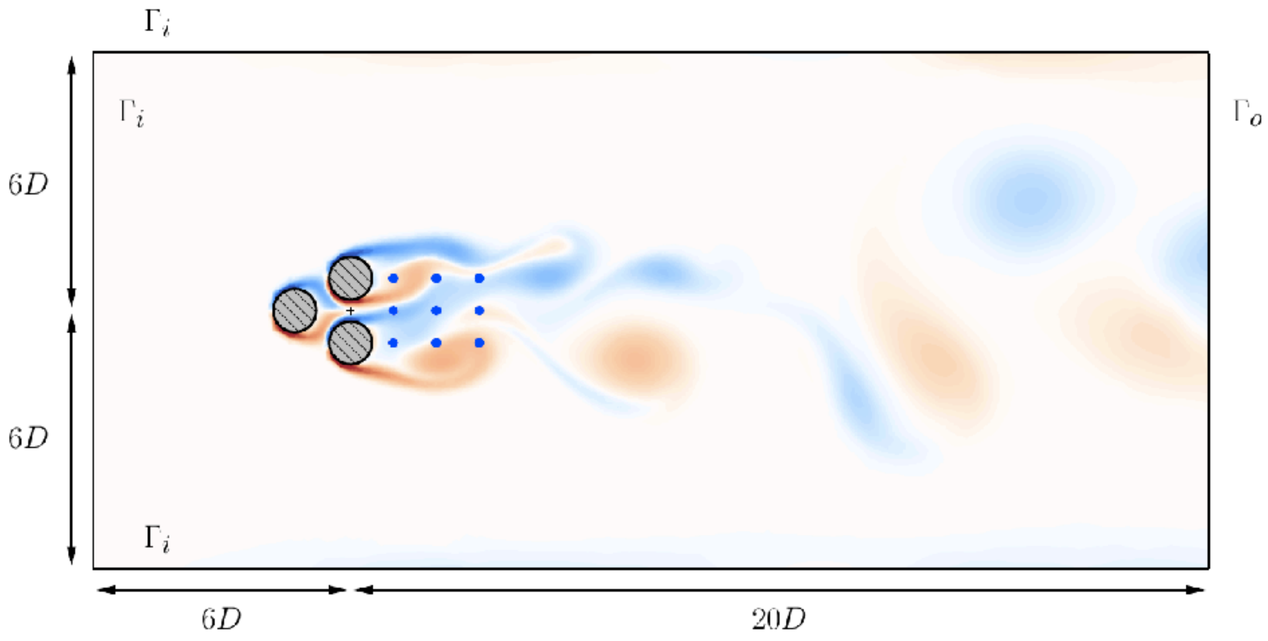
Deng Nan, et al. (2020), JFM

Fluidic Pinball case

$Re =$ [**100**, 150, 200]

Inflow: uniform velocity field

The fluidic pinball case (CFD)



Deng Nan, et al. (2020), JFM

Fluidic Pinball case

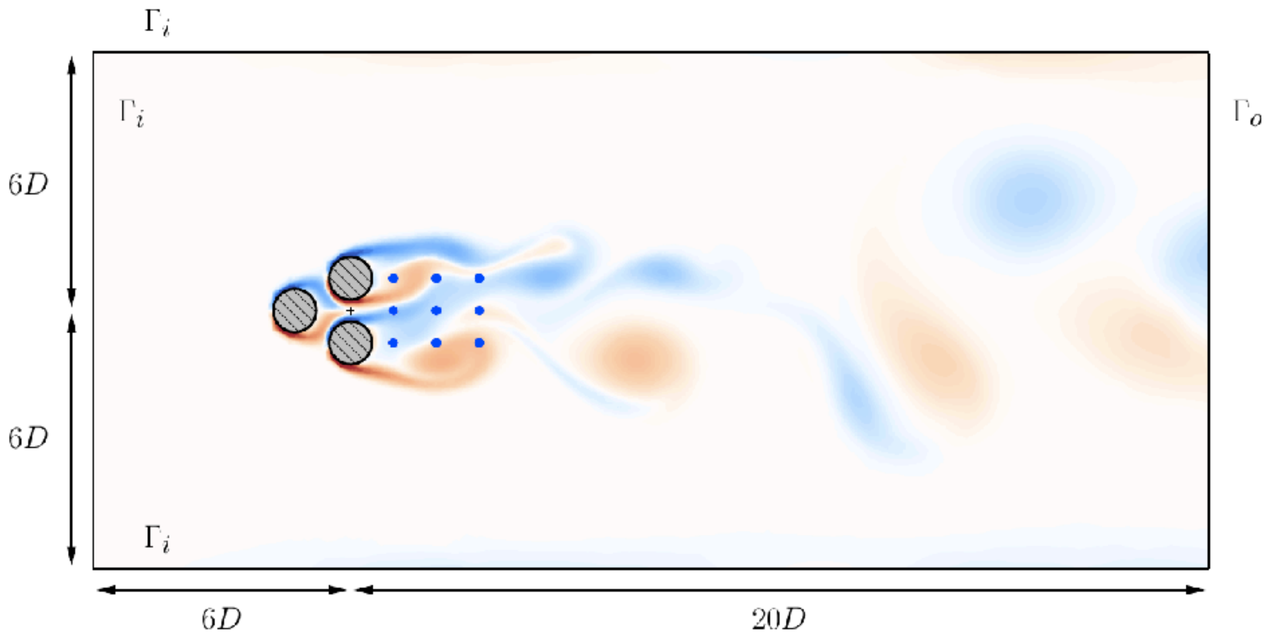
$Re =$ [**100**, 150, 200]

Inflow: uniform velocity field

State: **9 pressure** probes

Action: **angular velocity**

The fluidic pinball case (CFD)



Deng Nan, et al. (2020), JFM

Fluidic Pinball case

$Re =$ [**100**, 150, 200]

Inflow: uniform velocity field

State: **9 pressure** probes

Action: **angular velocity**

$$r(x, u) = -F_D^2 - \alpha \|a_t\|_2$$

$\alpha \in [0, 0.6]$ action **penalty**

$a_t \in [-3, 3]$ action **bounds**

TD3

The employed agent (i.e. learning algorithm) is the **Twin Delayed Deep Deterministic policy gradient**: TD3

$$TD = Q_\theta(x_n, u_n) - [r(x_n, u_n) + \max_u \gamma Q_{\theta'}(x_{n+1}, u)]$$

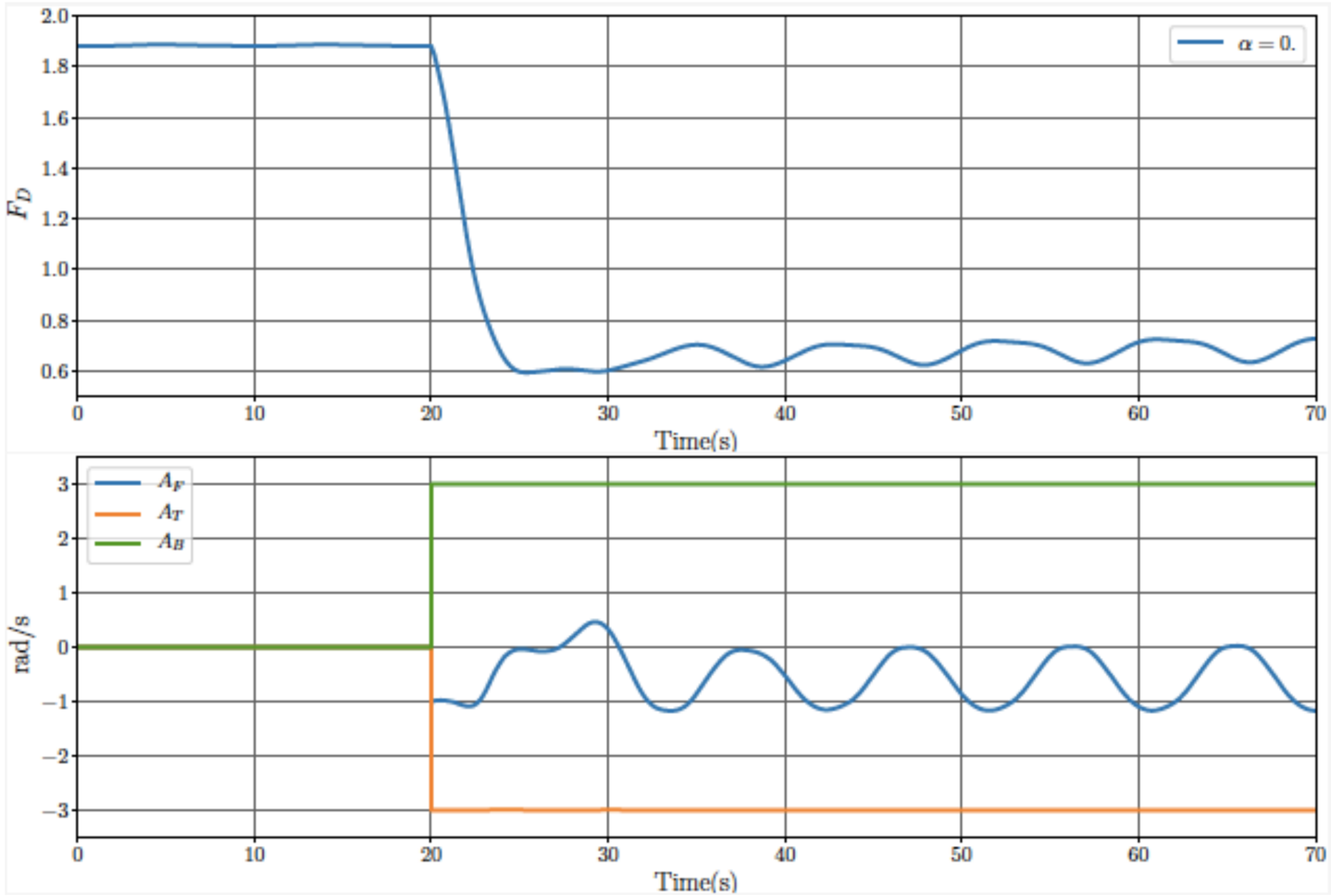
With a noisy Q function the $\max$ operation can **overestimate** its evaluation

- Clipped double Q-learning
- Delayed Policy Update

4 vortex shedding per epoch and 20 epochs are enough to converge the policy $\pi(x)$.

Fujimoto, Scott, Herke Van Hoof, and David Meger. "Addressing function approximation error in actor-critic methods." *arXiv preprint arXiv:1802.09477* (2018).

Results $\alpha = 0, 0.6$



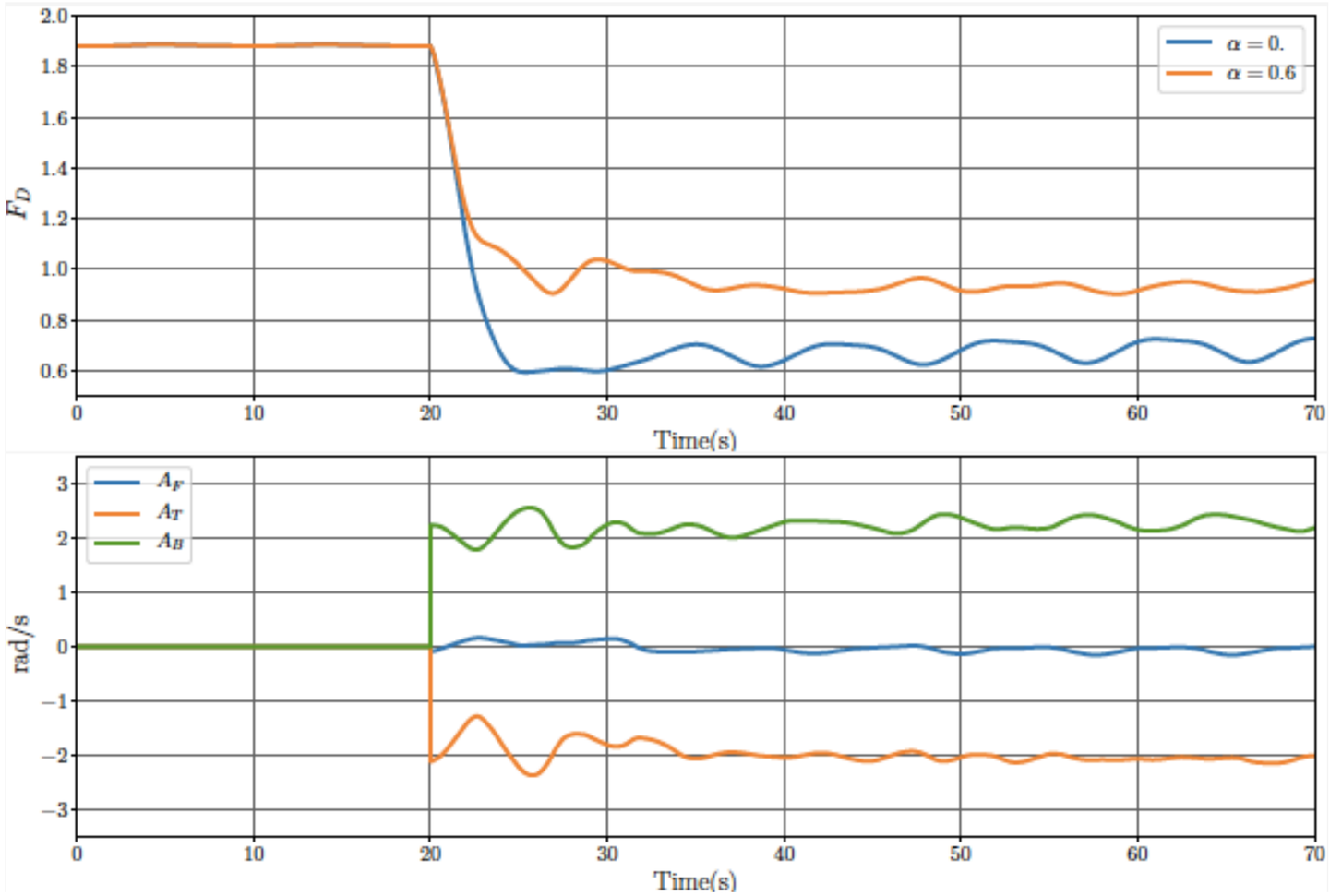
Without action penalty

$\alpha = 0$, 65 % drag reduction

Similar results obtained with **genetic algorithm**

Guy Y. Cornejo Maceda et al. (2021), JFM

Results $\alpha = 0, 0.6$



Without action penalty

$\alpha = 0$, 65 % drag reduction

Similar results obtained with **genetic algorithm**

With action penalty $\alpha = 0.6$

50 % drag reduction

Counter rotation of A_B and A_T

Guy Y. Cornejo Maceda et al. (2021), JFM

Low observability

With 9 probes:

- $\alpha = 0$: drag reduction $\approx 65\,\%$
- $\alpha = 0.6$: drag reduction $\approx 50\,\%$

With 3 probes:

- $\alpha = 0.6$: drag reduction $\approx 35\,\%$

With 1 probe:

- $\alpha = 0.6$: drag reduction $\approx 25\,\%$

Low observability

With 9 probes:

- $\alpha = 0$: drag reduction $\approx 65\,\%$
- $\alpha = 0.6$: drag reduction $\approx 50\,\%$

With 3 probes:

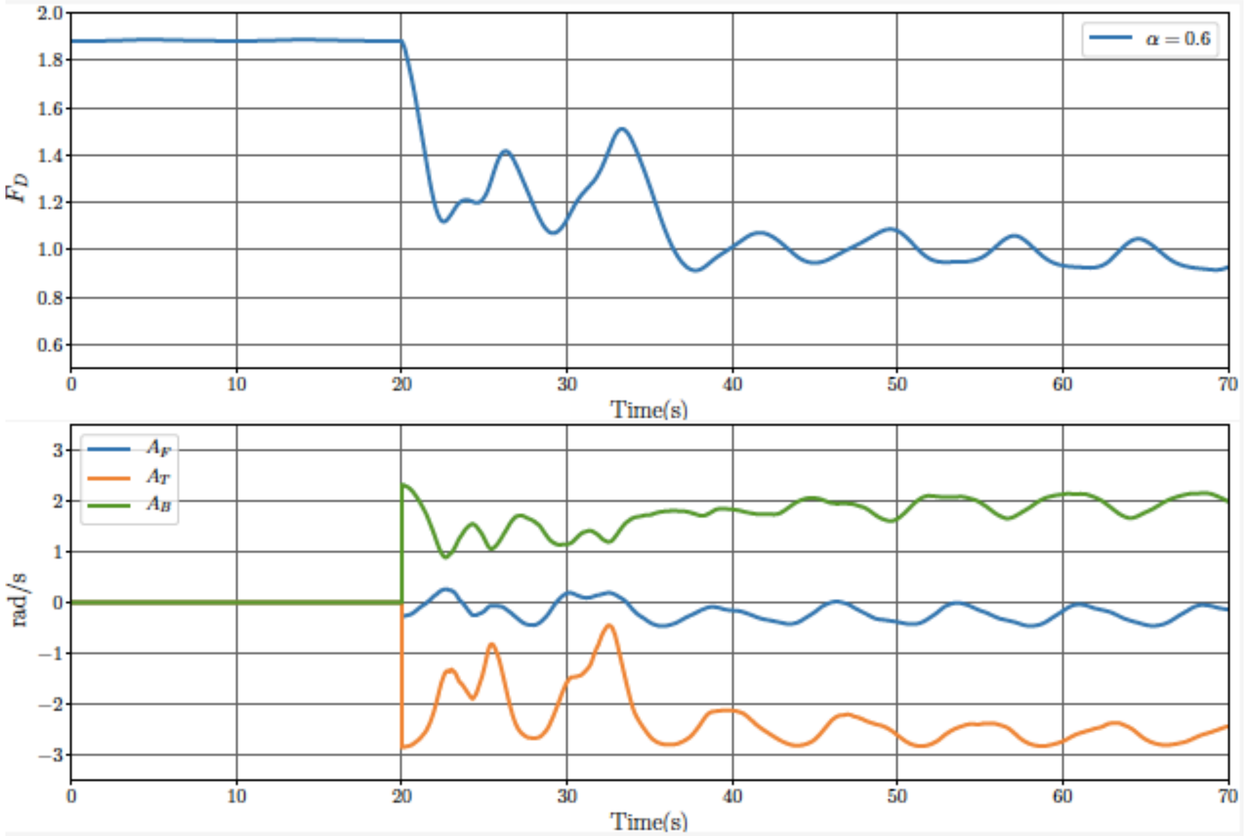
- $\alpha = 0.6$: drag reduction $\approx 35\,\%$

With 1 probe:

- $\alpha = 0.6$: drag reduction $\approx 25\,\%$

Reducing the number of probes the **full-observability hypothesis**, required by off-policy approach, is no longer satisfied → Introduce **temporal embedding**

Results $\alpha = 0.6$ + temporal embedding

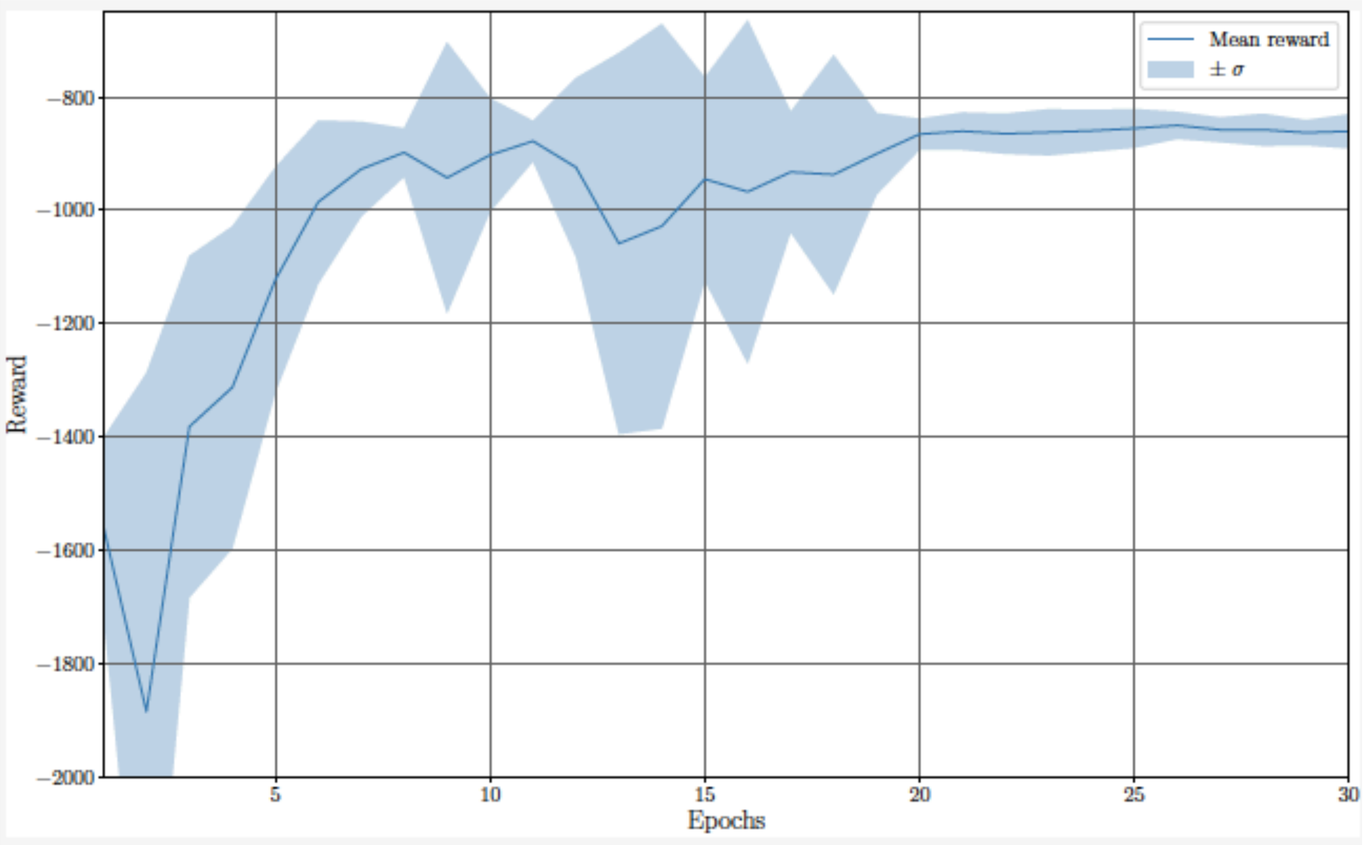


With 3 probes + temporal embedding (Taken's theorem):

- $\alpha = 0.6$: drag reduction $\approx \mathbf{50}\,\%$

as for the 9 probes penalised case

Results $\alpha = 0.6$ + temporal embedding



With 3 probes + temporal embedding (Taken's theorem):

- $\alpha = 0.6$: drag reduction $\approx$ **50 %**

as for the 9 probes penalised case

The learning is **robust** w.r.t. the initialisation of the Neural Network **parameters** θ and the **initial condition** x_0

Conclusions

- **Deep reinforcement learning** is a powerful method for **non-linear** control
  - **Full knowledge** of the system **is not required**
  - In principle, the policy is a **global optimum** if the cost function is solution of the Bellman equation
- Successfully tested on the **KS chaotic system** (PRSA, 2019)
- And on fluidic pinball case (APS 2020):
  - Different Re numbers tested
  - Fast training $\approx$ 4 vortex shedding per epochs and 20 epochs

Big dream: train an off-policy control agent with time delay (**credit assignment**)

- decoupling of the action and reward evaluation location in POMDP
- experimental open cavity

Optimality

Optimal control theory

Lev Pontryagin

x = state

u = action

$c(u, x)$ = cost $(-r(s, a))$

Dynamic programming

Richard Bellman

s = state

a = action

$r(s, a)$ = reward

Pontryagin maximum principle

To find the optimal policy it is necessary to solve the Hamiltonian system (i.e. variational problem on $[t_0, t_1]$) and a maximum condition of the Hamiltonian control.

Bellman's principle of optimality:

An optimal policy has the property that whatever the initial state and initial decision are, the remaining decision must constitute an optimal policy with regard to the state resulting from the first decision.

According to Pontraygin

We have to solve the *variational problem*

$$J = \int_0^T r(x(t), u(t))dt \quad \text{s.t.} \quad \dot{x} = f(x, u)$$

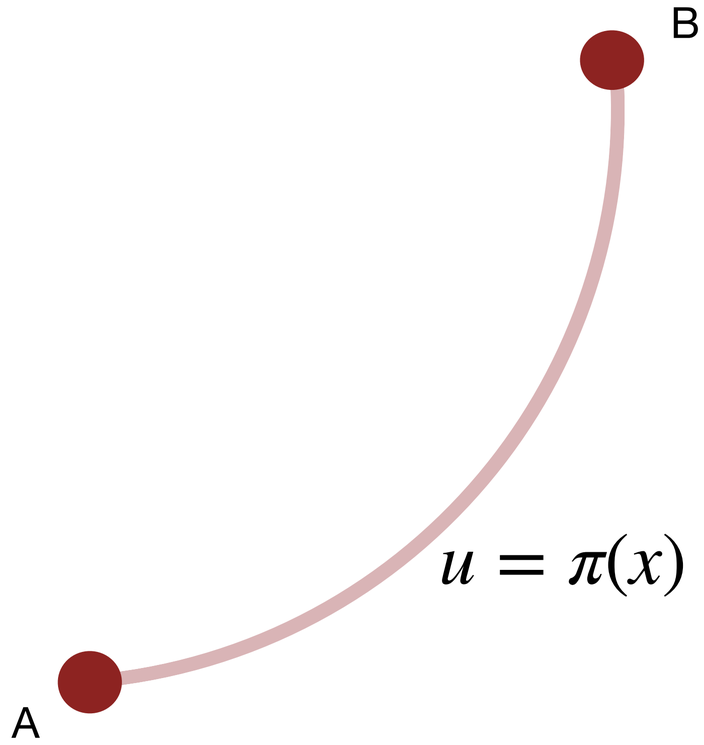
Introducing the Lagrangian multiplier $\lambda(t)$

$$H(x, u, \lambda, t) = \lambda^{\mathrm{T}} f(x, u) + r(x, u)$$

we can solve:

1. $H_\lambda = \dot{x} = f(x, u)$
2. $H_x = -\dot{\lambda} = \lambda^T f_x(x, u) + r_x(x, u)$
3. $H_u = 0 \Rightarrow u^* = \pi^*(x^*)$

It is a **necessary** condition for optimality



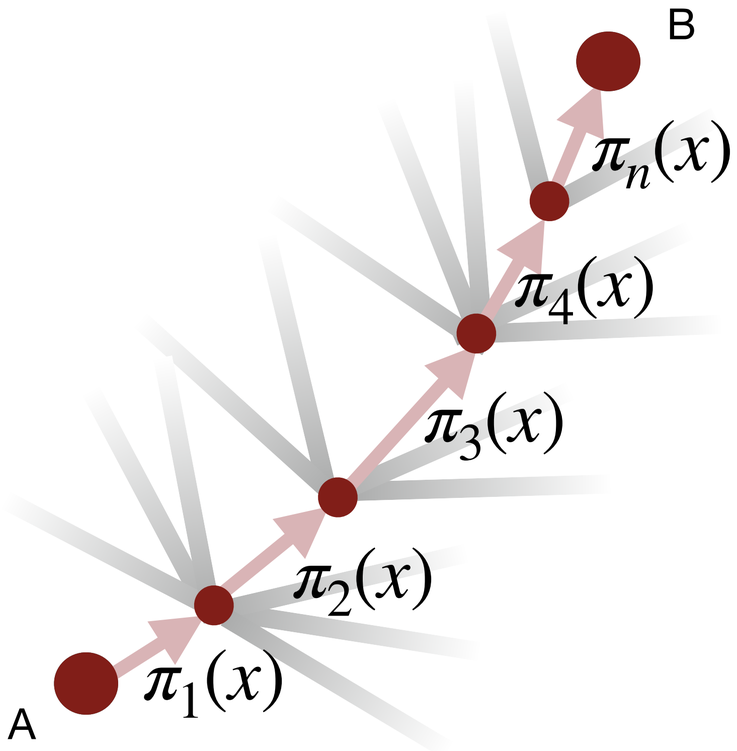
According to Bellman

Given $V(x,t)$ with $x \in \mathbb{R}^n$, $V : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}$ solution of the Hamilton-Jacobi-Bellman equation,

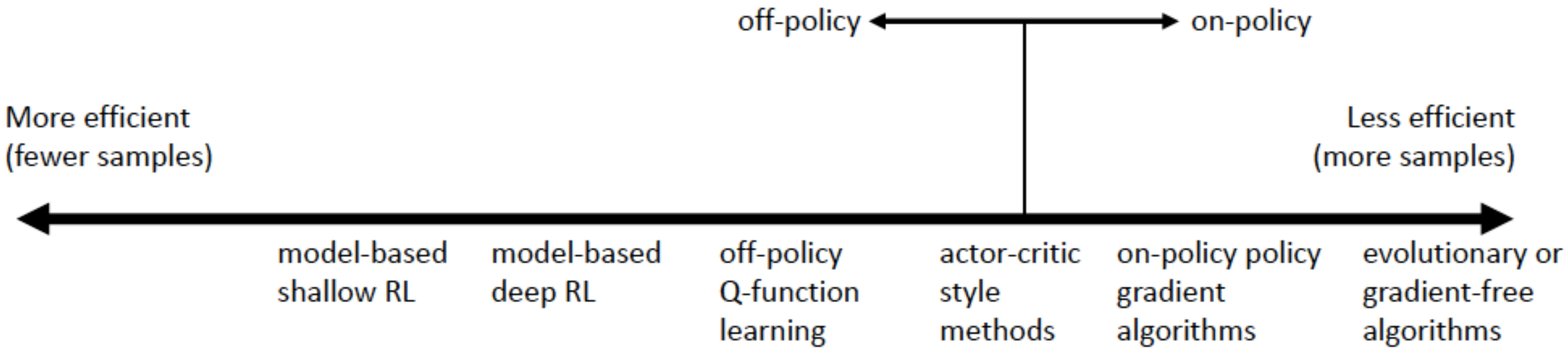
$$-V(\dot{x},t) = \max_u (\nabla V(x,t) \cdot f(x,u) + r(x,u))$$

the optimal action u^* at x_t is the one that maximise $V(x_{t+1})$

It is a **sufficient** and **necessary** condition for optimality



Comparison



Picture from Levine's DRL class